ETH Zürich

Towards Gigahertz Bandwidth Digital Signal Processing in Circuit Quantum Electrodynamics

MASTER THESIS of

Yves Salathé

SUPERVISION

Prof. Dr. Andreas Wallraff

October 7, 2011

The objective of circuit quantum electrodynamics (circuit QED) is to study the quantum mechanical interaction between artificial atoms realized in superconducting circuits with the electromagnetic field of a resonator. This approach is particularly interesting since it obeys the same quantum mechanical description as a cavity QED system where an atom interacts with the light confined in a cavity.

Digital signal processing has become an important tool in circuit QED research. In particular the ability to digitally compute correlation functions of measured RF signals can be used to test quantum mechanical predictions for circuit QED systems which were originally only verified in the optical frequency domain.

In digital signal processing, the rate at which the signal is measured determines the width of the frequency band about which information can be gained. It is thus an important challenge to push the sampling and processing rate as high as possible. Field programmable gate arrays (FPGA) offer not only high flexibility in digital signal processing but can also process digital signals at high rates due to their inherent parallel architecture.

The subject of the present thesis is the design and demonstration of an FPGA-based measurement instrument for circuit QED capable of measuring and processing signals with a bandwidth of 500 MHz.

Using this instrument, we measure the power spectrum of the resonance fluorescence originated from a transmon qubit coupled to a coplanar waveguide resonator. The spectrum clearly shows a Mollow triplet structure with a center peak and two satellite peaks.

The data fits well to the theoretically predicted Mollow triplet of a two-level system for drive powers that correspond to Rabi frequencies below $\Omega_\mathrm{R} \approx 40\,\mathrm{MHz}$. However, when the parameters obtained by this fit are applied to extrapolate the spectrum for higher drive powers, there is a significant deviation from the measured data which indicates that the two-level approximation is not feasible for large drive powers.

Contents

1 Introduction

Circuit quantum electrodynamics (circuit QED) is the field in physics with the goal to investigate the quantum mechanical interaction of atom-like systems with the electromagnetic field of a resonator realized in a superconducting circuit [1]. The quantum mechanical description of these systems reveals a close analogy to cavity QED [2] where light at optical frequencies is confined between two mirrors and interacts with atoms that are located in the volume of the cavity as described in Section 2.1.

In the same manner as cavity QED gives insight into the light-matter interaction at optical frequencies, circuit QED makes it possible to observe quantum mechanical interaction taking place at microwave frequencies on a superconducting chip. In particular, purely quantum mechanical effects involving single photons at microwave frequencies were observed in superconducting circuits [3–6] which were originally only known in the optical frequency domain.

The resonator also makes it possible to mediate the interaction between multiple artificial atoms, so called "quantum bits" (qubit) [7–12]. Thus circuit QED is a promising approach for the long awaited implementation of quantum algorithms.

Measurements in circuit QED are usually performed by sending a probe microwave tone to the resonator and measuring the transmitted or reflected signal. For the measurement, the signal has to be amplified and down-converted so that the time-dependent voltage of the electromagnetic field can be digitized using an analog-to-digital converter. The digital signal can then be processed in order to extract the relevant information about the circuit QED system from the measured signal.

Furthermore the possibility to control the quantum system by generating feedback signals from the measured signals has been recently investigated in the circuit QED research community. Feedback signals could be e.g. used to stabilize entangled states as described in [13] or to implement teleportation schemes [14]. However, the technical challenge that has to be solved for such applications is to generate the feedback within the time the quantum system remains coherent which is for circuit QED systems typically on the order of 1 µs.

While single microwave photon detection is under investigation [15–19], the measurement of field correlation functions [20–22] can be used to test the predictions of quantum optics. The computation of these correlation functions from the data points (samples) that have been obtained for the time-dependent voltage with high resolution is challenging, since it involves averaging over many evaluations of the correlation function.

Due to the high amount of data points needed, it is beneficial to perform the computation of the correlation function in real-time which means to perform the function evaluation and averaging at the rate with which the data is sampled instead of storing a large amount of raw data to a digital memory. The problem is that conventional central processing units (CPU) as available in personal computers execute programs in a sequential manner and are thus usually not fast enough to process a digital signal that has been sampled at a rate of e.g. 1 GHz with a precision of e.g. 12 bits per sample. This is because a single processing unit clocked at e.g. 3 GHz could only perform a maximum of 3 operations on every data point until the next data point had to be processed.

In contrast, field programmable gate arrays (FPGA) offer an inherent parallel architecture which makes it possible to receive and process in every clock cycle a number of data points in parallel.

In the present thesis we design an FPGA based measurement instrument with the ability to digitally compute the power spectrum of the signal at a sampling rate of 1 billion samples per second. Due to the Nyquist theorem [23], this sampling rate corresponds to a maximal aliasing-free bandwidth 500 MHz.

To demonstrate the feasibility of using this instrument in circuit QED, we measure the power spectrum of the microwave radiation scattered by the quantum mechanical system of an artificial atom coupled to a resonator.

2 An Introduction to Circuit QED

This chapter contains a brief introduction to the physics that is studied with the measurement instrument developed in the present thesis. The review follows a bottom up approach beginning with one of the two basic elements of circuit QED: the artificial atom described in Sections 2.2 and 2.3. The other basic element is the coplanar waveguide resonator to which the artificial atom is coupled. This coupled system is described in Section 2.4.

Going up one level of abstraction, we discuss the model which establishes the connection to quantum optics in general and in particular to cavity QED in Section 2.5. After that, we discuss the theory behind the resonance fluorescence measurement presented in Sections 2.6 and 2.7.

2.1 Analogy to Cavity QED

In the following, a short overview of a typical cavity QED setup is given in order to highlight the analogies to circuit QED. Some terminology of this section will be used when circuit QED is introduced in Section 2.

Figure 1 shows the basic setting of a cavity QED experiment schematically. The electric dipole of an atom interacts with the light confined between two mirrors (cavity) which leads to vacuum Rabi oscillations where a photon of a specific mode of the electromagnetic field inside the cavity, also called fundamental mode, is repeatedly absorbed and re-emitted by the atom at rate g. This setting is described by the so called Jaynes-Cummings model of quantum optics [24] which is described in more detail in Section 2.5.

As in reality such a system cannot be perfectly isolated, there is a non-zero probability that the atom emits photons to other modes than the fundamental mode of the cavity. Likewise energy can leak out of the fundamental mode to the environment instead of to the atom. These two decays are usually described by the decay rates γ and κ respectively.

Note that there is a trade-off between trying to keep the photon decay rate κ as small as possible and providing a way to couple the cavity to an external field through an input/output port such as a semi-transparent mirror. Sometimes the photon decay rate is also expressed in terms of the quality factor $Q = \omega_\mathrm{r}/\kappa$, where ω_r is the frequency of the fundamental mode.

2.2 Cooper Pair Box

The goal of the following discussion is to explain the artificial atoms used in the circuit QED experiments performed in the present thesis.

The Cooper pair box [25, 26] comprises two superconducting reservoirs that are separated by one or usually two Josephson junctions. Josephson junctions are formed by a thin layer of insulating material, usually an oxide layer, between two superconducting electrodes. Due to quantum mechanical tunneling, a current can flow through the junction.

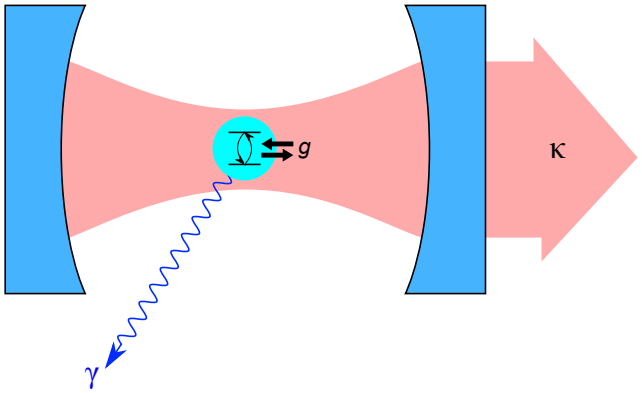


Figure 1: Conventional illustration of a typical cavity QED experiment [7]. Light is confined between the two blue concave shaped mirrors. The cyan colored circle in the center between the two mirrors represents a two-level system which can be an atom. The interaction of the cavity field with the two-level system takes place at rate g. Light escapes the cavity through the semi-transparent mirror at rate κ. The blue wave represents radiation the atom emits into other modes than the cavity fundamental mode at rate γ.

For superconductors it is possible to describe the quantum mechanical state of the electron gas inside an electrode by a single macroscopic wave function. Electrons in the superconductor form so called Cooper pairs. Josephson discovered that the tunneling of these pairs of electrons across the junction can be described in terms of the phase difference of a macroscopic wave function of the two electrodes [27–29].

In the traditional Cooper pair box, one of the electrodes is much smaller and is thus called the island. Through the Josephson junction, Cooper pairs can tunnel from and to the island at the expense of the so called Josephson energy E_J. For the quantum mechanical description of this system the charge basis can be used which is formed by the states $|N\rangle$, corresponding to the number of additional Cooper pairs on the island compared to the number of Cooper pairs present when the charges on the island are neutralized. The Hamiltonian that describes the tunneling of Cooper pairs from and to the island can be written in the charge basis as follows [30]

$$\hat{H}_\mathrm{J} = \frac{E_\mathrm{J}}{2} \sum_N \left(|N\rangle\langle N+1| + |N+1\rangle\langle N| \right). \tag{1}$$

With an additional capacitor, called the gate capacitor with capacitance C_g, which connects the island to a voltage source at voltage V_g, one can set a voltage offset that can cause Cooper pairs to tunnel through the Josephson junction from or to the island. The corresponding circuit is shown in part (a) of Figure 2. The Hamiltonian that describes the effect of the gate capacitor is [26]

$$\hat{H}_\mathrm{el} = E_\mathrm{C} \sum_N (N - n_\mathrm{g})^2 \, |N\rangle\langle N|, \tag{2}$$

where $n_g = C_g V_g/(2e)$ is the gate induced polarization charge in units of Cooper pairs. The charge of a single electron is denoted by e. The prefactor E_C is the energy needed to transfer a Cooper pair onto the island

$$E_C = \frac{(2e)^2}{2C_\Sigma}, \tag{3}$$

where $C_\Sigma = C_g + C_J$ is the total capacitance of the island. Note that C_J in the denominator is the inherent capacitance of the Josephson junction. In a real experiment there are likely other stray capacitances to be taken into account when calculating E_C.

The Hamiltonian which describes the circuit of the Cooper pair box is then given by the sum of $\hat{H}_J$ and $\hat{H}_{el}$

$$\hat{H}_{CPB} = \hat{H}_J + \hat{H}_{el}. \tag{4}$$

The time-independent Schrödinger equation for $\hat{H}_{CPB}$ can be solved numerically or even analytically by transforming the Hamiltonian into its phase basis representation [31]. Note that phase is the variable conjugate to charge. Figure 2 (b) shows the energy level diagram of the Cooper pair box as a function of n_g for the ground state and the first two excited states.

It can be clearly seen that for most choices of n_g, the spacing $E_{01} = E_1 - E_0$ between the energy levels of the ground state E_0 and the first excited state E_1 differs from the spacing between the energy levels of the first and second excited state $E_{12} = E_2 - E_1$. The difference $E_{12} - E_{01}$ is called anharmonicity. If the anharmonicity is non-zero, then the energy spectrum of the system differs from the one of a harmonic oscillator for which the energy levels are equally spaced.

Supposing the system is in its ground state, the absorption of a photon of frequency $\nu_{01} = E_{01}/h$ will drive the system into its first excited state. If the system is anharmonic, the system will not absorb a photon of frequency ν_{01} when it is in the excited state. For a large enough anharmonicity it is thus valid to treat the Cooper pair box as an effective two-level system, realizing a so called quantum bit (qubit).

When island and ground are connected by two Josephson junctions instead of only one, the system is called a "split" Cooper pair box [30]. The circuit representing the split Cooper pair box is displayed in of Figure 2 (c).

As long as there is no magnetic flux threading the loop formed by the two Josephson junctions, the split Cooper pair box is described by the same Hamiltonian as for the single Josephson junction given by Equation (4) with the Josephson energy E_J being the sum of the Josephson energies $E_{J,1}$ and $E_{J,2}$ of the individual Josephson junctions

$$E_J = E_{J,1} + E_{J,2}. \tag{5}$$

If a magnetic flux Φ threads the loop formed by the two Josephson junctions, then the Hamiltonian of Equation (1) which describes the coupling induced by the Josephson junctions changes as the superconducting phase across the junctions becomes flux dependent. However, if the two junctions have the same Josephson energy, i.e. if $E_{J,1} = E_{J,2}$, one can define an effective Josephson energy [30]

$$E_J^{\text{eff}} = \cos\left(\pi \frac{\Phi}{\Phi_0}\right) 2E_{J,1}, \tag{6}$$

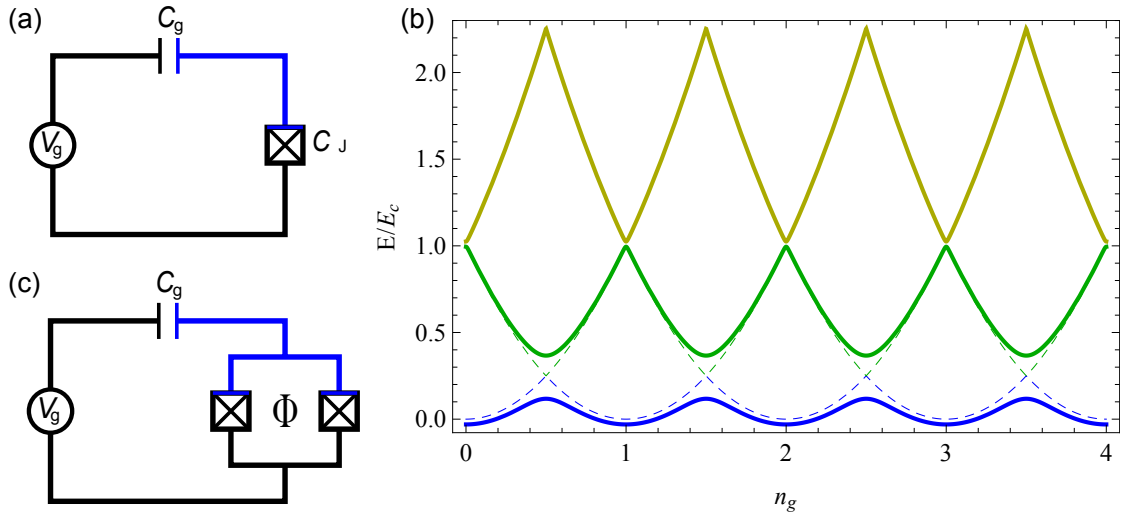


Figure 2: **(a)** Circuit diagram of a Cooper pair box. The box with the cross inside represents the Josephson junction with capacitance C_J. The gate capacitor is labeled by its capacitance C_g and the gate voltage source by its voltage V_g. The wire colored in blue is called the island. **(b)** Energy level diagram of the Cooper pair box as a function of the polarization charge n_g for $E_J = E_C/4$. The lowest three energy levels are shown. The dashed lines indicate the energy diagram for $E_J = 0$. **(c)** Circuit diagram of a split Cooper pair box with magnetic flux Φ threading the loop.

which can be substituted into Equation (1) in place of E_J. Note that $\Phi_0 = h/2e$ is the flux quantum and h is Planck's constant. From Equation (6), it can be seen that the split Cooper pair box allows to tune the Josephson energy and hence the splitting between the energy levels by applying an external magnetic flux to the loop. This is convenient as it allows to define the transition energy and hence the frequency at which the qubit can absorb and emit photons.

2.3 Transmon

One major problem with the standard implementation of the Cooper pair box as presented in Section 2.2 is that for small ratios $E_\mathrm{J}/E_\mathrm{C}$, the energy bands displayed in Figure 2 (b) respond strongly to fluctuations of the bias charge n_g. Thus the different eigenstates may acquire random phases. This so called dephasing should be suppressed as much as possible in order to observe quantum effects on long timescales.

One approach to make the Cooper pair box more resilient to charge noise is to operate it at the so called charge sweet spot which means to select a value of half a Cooper pair for n_g where the dependence of the energy of the ground and first excited state is flat, see Figure 2 (b).

Another approach is to make the ratio $E_\mathrm{J}/E_\mathrm{C}$ large since the slope of the curves displayed in part (b) of Figure 2 decreases as the ratio $E_\mathrm{J}/E_\mathrm{C}$ is increased. The idea behind the transmon is to increase the ratio $E_\mathrm{J}/E_\mathrm{C}$ by adding an additional capacitor with large capacitance C_B in parallel to the Josephson junctions of the split Cooper pair box [32]. By this modification, the charging energy E_C given by Equation (3) becomes small compared to E_J as the total capacitance $C_\Sigma = C_\mathrm{g}+C_\mathrm{J}+C_\mathrm{B}$ contains the additional large capacitance C_B. It is shown in [32] that the Hamiltonian of the transmon can be formulated identical to the Hamiltonian of the Cooper pair box given in Equation (4). Increasing the ratio $E_\mathrm{J}/E_\mathrm{C}$ comes at the expense of reduced anharmonicity. However since the anharmonicity is still non-zero, one can still operate the transmon with the two-level approximation if enough care is taken not to excite any higher excited states.

2.4 Coupling a Transmon to a Coplanar Waveguide Resonator

In this section it is shown how a superconducting qubit, in particular the transmon can be coupled to a coplanar waveguide resonator in order to create the setting of a circuit QED experiment: an artificial atom couples to the electromagnetic radiation in a cavity realized as a superconducting circuit.

Coplanar waveguide resonators follow the same principle as coaxial cables which are used for the transmission of radio frequency (RF) signals [30]. An isolated center conductor is surrounded by a ground plane which acts as shield so that electromagnetic waves are guided along the line formed by center conductor and shield. The major difference of coplanar waveguide resonators compared to coaxial cables is that the coplanar waveguide resonators are only shielded by ground planes along one direction. It is called coplanar waveguide since the ground planes as well as the center conductor are approximately two dimensional objects that live all in the same plane. The waveguide

is turned into a resonator by impedance mismatches at both ends in longitudinal direction. The impedance mismatches reflect electromagnetic waves which then eventually form standing waves along the lines between the two impedance mismatches.

The coplanar waveguide resonator, as schematically depicted in Figure 3, is made by inserting insulating gaps into a layer of superconducting metal that has been deposited on a chip. Typically, the thickness of the layer is as small as 150 nm. The center conductor of the coplanar waveguide resonator is separated by an insulating gap of typically 5 µm from the ground planes on each side in lateral direction [33]. The width of the center conductor in the lateral direction of the conductor is usually 10 µm.

In order to turn the coplanar waveguide into a resonator, the center conductor is truncated to finite length L by interrupting the conductor with two capacitors on each end that couple the resonator to the input/output lines. The length L is at least 3 orders of magnitude larger than the width of the center conductor. With these geometric properties the mode frequencies of the coplanar waveguide resonator are given by

$$\omega_k = \frac{k\pi}{L}\sqrt{\frac{1}{lC}}, \qquad k \in \mathbb{N}, \tag{7}$$

where l denotes the inductance per unit length, C is the capacitance per unit length and $\mathbb{N}$ is the set of natural numbers.

If the wave inside the coplanar waveguide resonator has a wavelength much longer than the diameter of the waveguide's cross-section, then the coplanar waveguide resonator can be approximated as an infinite series of inductors where to each node between two inductors a capacitor that connects to the ground is attached [7]. This approximation is illustrated in the circuit diagram of Figure 3.

The transmon is placed in the gap between the center conductor and the ground plane of the coplanar waveguide resonator so that one gate capacitor with capacitance C_{g_1} is formed by the insulator between the transmon's upper island and the center conductor of the resonator as well as a gate capacitor with capacitance C_{g_2} between the ground plane and the lower island as shown in the circuit diagram of Figure 3.

When the transmon is located in the middle of the resonator, it can couple to the $k = 2$ resonator mode which has an anti-node at the resonator's longitudinal center meaning that the root mean square voltage V_{rms} of the electromagnetic radiation with frequency ω_2 is maximal at the transmon's position. In the following we denote the frequency of the resonator mode to which the transmon couples as ω_{r}.

The resonator mode under consideration can be modeled by a LC oscillator which is represented by the red colored part of the circuit diagram in Figure 4. The rest of the circuit shown in Figure 4 is an equivalent circuit for the more complicated circuit that takes into account all capacitances involved to accurately represent the transmon inside the coplanar waveguide resonator as discussed in a paper by J. Koch et al. that introduces the transmon [32]. The effective capacitances C_{g}, C_{B} and C_{in} in the equivalent circuit are obtained by applying Thévenin's theorem on the complex impedances of the capacitors.

Quantum mechanically, the LC oscillator can be described as a harmonic oscillator. We write the Hamiltonian for the harmonic oscillator in terms of creation and annihila-

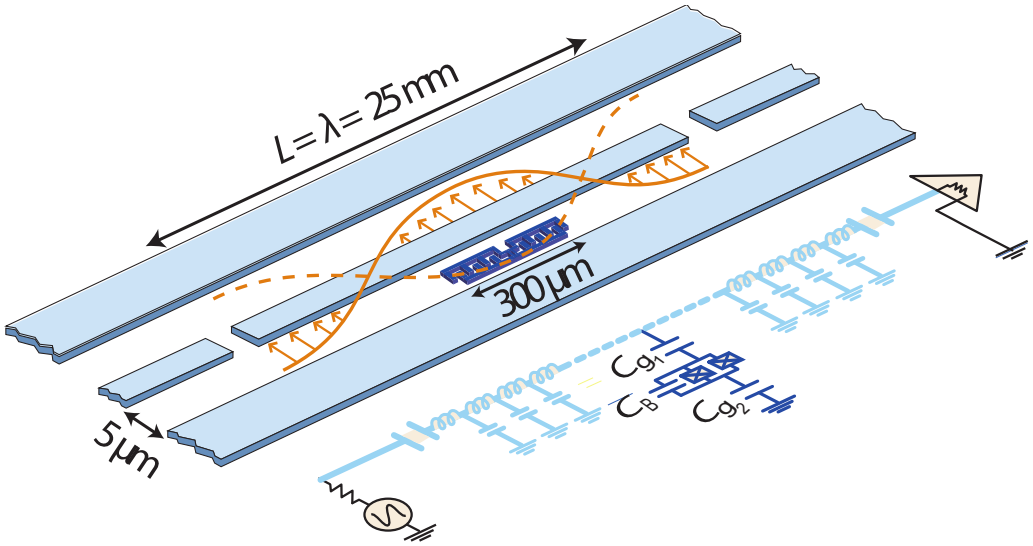


Figure 3: Schematic picture of a coplanar waveguide resonator with a transmon positioned inside the gap between center conductor and ground. The light blue boxes indicate the layers of superconducting metal that form the conducting part of the coplanar waveguide resonator which consists of a center conductor surrounded by two ground planes in lateral direction and two capacitively coupled transmission lines in longitudinal direction. The orange wave indicates the voltage of the standing wave for the $k = 2$ mode of the resonator at its maximal deflection. The dark blue structure centered inside the gap between the center conductor and the lower ground plane is the artificial atom called transmon. Below the schematic a circuit diagram is shown which approximatively describes the system of the transmon coupled to the coplanar waveguide resonator. Figure adapted from [34].

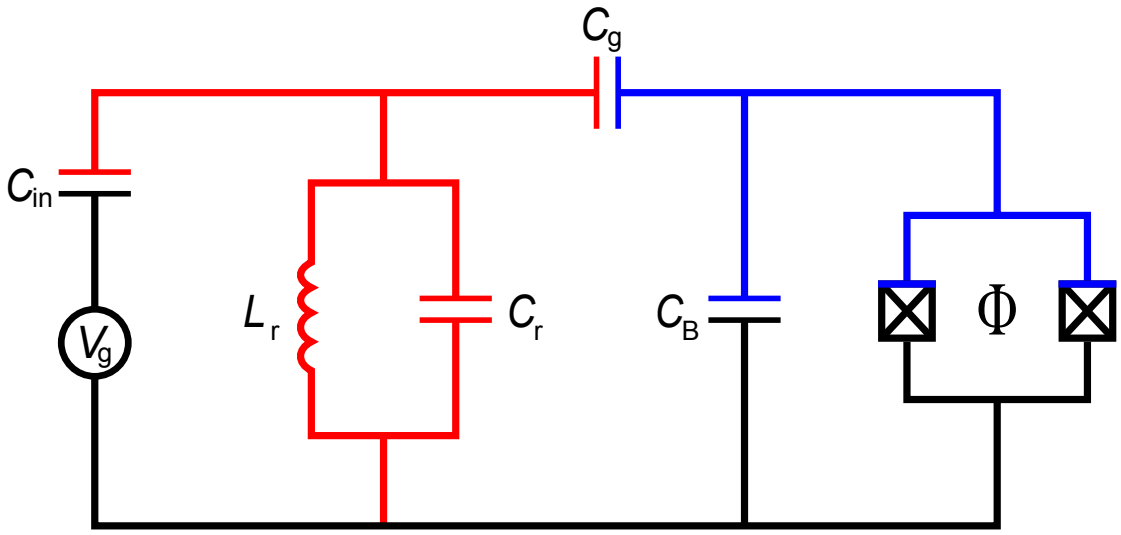


Figure 4: Simplified circuit to model the interaction between the transmon and the coplanar waveguide resonator. The red part of the circuit depicts an LC oscillator with total capacitance C_r and total inductance L_r representing the resonator mode. The capacitances $C_\mathrm{in}, C_\mathrm{g}$ and C_B are effective capacitances that have to be derived from the real capacitances by Thévenin's theorem.

tion operators $\hat{a}^\dagger$ and $\hat{a}$ which create or annihilate one excitation of the oscillator

$$\hat{H}_\mathrm{r} = \hbar\omega_\mathrm{r}\left(\hat{a}^\dagger\hat{a} + \frac{1}{2}\right). \tag{8}$$

The gate voltage seen by the transmon depends on the electric field of the resonator mode

$$\hat{V} = V^0_\mathrm{rms}\left(\hat{a} + \hat{a}^\dagger\right), \tag{9}$$

where $V^0_\mathrm{rms} = \sqrt{\hbar\omega_2/2C_\mathrm{r}}$ is the root mean square voltage on the center conductor due to vacuum fluctuations.

The Hamiltonian of the combined system contains the uncoupled Hamiltonians $\hat{H}_\mathrm{CPB}$ and $\hat{H}_\mathrm{r}$ given in Equations (4) and (8) as well as a coupling term which depends on the gate voltage operator $\hat{V}$ given in Equation (9). The Hamiltonian is written as [32]

$$\hat{H} = \hat{H}_\mathrm{CPB} + \hat{H}_\mathrm{r} + 2e\beta\hat{n}\hat{V} \tag{10}$$

where $\beta = C_\mathrm{g}/C_\Sigma$ denotes the ratio of the effective gate capacitance and the total capacitance. The operator $\hat{n}$ represents the number of Cooper pairs on the upper island of the transmon which is in the charge basis written as $\hat{n} = \sum_N N|N\rangle\langle N|$. Note that $\hat{H}_\mathrm{CPB}$ contains a constant gate voltage that corresponds to the DC voltage between the center conductor and the ground plane.

2.5 Jaynes Cummings Model

If the Hamiltonian of the combined system given by Equation (10) is written in the basis of eigenstates $|j\rangle$ of the Hamiltonian $\hat{H}_{\mathrm{CPB}}$ from Equation (4) that describes the transmon subsystem alone, one obtains a generalized form of the so called Jaynes-Cummings Hamiltonian [32]

$$\hat{H} = \hbar \sum_j \omega_j^{\mathrm{atom}} |j\rangle\langle j| + \hbar\omega_{\mathrm{r}} \left(\hat{a}^\dagger \hat{a} + \frac{1}{2} \right) + \hbar \sum_{i,j} g_{ij} |i\rangle\langle j| \left(\hat{a} + \hat{a}^\dagger \right), \tag{11}$$

where $\hbar\omega_j^{\mathrm{atom}}$ is the eigenvalue that corresponds to eigenstate $|j\rangle$ of the uncoupled Hamiltonian of the transmon. The coupling coefficients are given by

$$g_{ij} = \frac{2\beta e V_{\mathrm{rms}}^0}{\hbar} \langle i|\hat{n}|j\rangle. \tag{12}$$

The Jaynes-Cummings Hamiltonian was first derived in quantum optics and describes a general cavity QED experiment where the electric dipole formed by a single atom interacts with one of the modes of confined electromagnetic radiation. To derive the Jaynes-Cummings Hamiltonian in its original form, the (artificial) atom is approximated as a two-level system. Furthermore the rotating wave approximation is employed, which means to neglect fast oscillating interactions. Using these approximations, the Jaynes-Cummings Hamiltonian from Equation (11) becomes

$$\hat{H} = \frac{\hbar\omega_{\mathrm{a}}}{2} \left(|1\rangle\langle 1| - |0\rangle\langle 0| \right) + \hbar\omega_{\mathrm{r}} \left(\hat{a}^\dagger \hat{a} + \frac{1}{2} \right) + \hbar g_{01} \left(\hat{a}^\dagger |0\rangle\langle 1| + |1\rangle\langle 0| \hat{a} \right), \tag{13}$$

where $\omega_{\mathrm{a}} = \omega_1^{\mathrm{atom}} - \omega_0^{\mathrm{atom}}$ is the transition energy from the ground state to the excited state of the qubit.

In the following, we consider the Jaynes-Cummings model for the case in which the resonator mode is in resonance with the transition frequency of the lowest two levels of the atom, i.e. when $\omega_{\mathrm{a}} - \omega_{\mathrm{r}} \ll g$.

The eigenstates of the combined system that correspond to the split energy levels are superpositions of two states of the uncoupled system where one excitation is either in the qubit or in the resonator [30]

$$|n\pm\rangle = \frac{1}{2} \left(|n\rangle|\mathrm{g}\rangle \pm |n-1\rangle|\mathrm{e}\rangle \right). \tag{14}$$

In order to distinguish the eigenstates of the qubit from the eigenstates of the resonator we denote the ground and excited states of the qubit by $|\mathrm{g}\rangle$ and $|\mathrm{e}\rangle$. Due to the dipole coupling the system of qubit and resonator can no longer be treated separately: the bare states of the uncoupled system turned into "dressed" states $|n\pm\rangle$. The spectrum of these dressed states is given by

$$E_{n\pm} = \hbar \left(n\omega_{\mathrm{r}} \pm \sqrt{n} g \right). \tag{15}$$

The spacing of $2\sqrt{n}g$ between the energy levels is called vacuum Rabi mode splitting. Note that the dressed energy level spectrum inherits the non-linearity of the atom energy levels so that the combined system of qubit (artificial atom) and resonator mode (cavity) is sometimes also called "atom-cavity molecule" [35].

2.6 Correlation Functions in Circuit QED

In order to analyze the electromagnetic radiation that exits the resonator, methods from quantum optics can be used. The measurement of correlation functions is a commonly used technique in the optical frequency domain.

The first-order autocorrelation function of a complex signal $S(t)$ is defined as

$$G^{(1)}(\tau) \equiv \int_{-\infty}^{\infty} S(t)^* S(t+\tau)\, dt. \tag{16}$$

The Fourier transform of the first-order autocorrelation function is calculated as follows:

$$\begin{aligned}
\mathcal{F}[G^{(1)}(\tau)](\nu) &= \int_{-\infty}^{\infty} G^{(1)}(\tau)\, \mathrm{e}^{-2\pi i\nu\tau}\, d\tau \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} S(t)^* S(t+\tau)\, dt\, \mathrm{e}^{-2\pi i\nu\tau}\, d\tau \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} S(t)^* S(\tilde{\tau})\, dt\, \mathrm{e}^{-2\pi i\nu(\tilde{\tau}-t)}\, d\tilde{\tau} \\
&= \left(\int_{-\infty}^{\infty} S(t)^*\, \mathrm{e}^{2\pi i\nu t}\, dt\right)\left(\int_{-\infty}^{\infty} S(\tilde{\tau})\, \mathrm{e}^{-2\pi i\nu\tilde{\tau}}\, d\tilde{\tau}\right) \\
&= \left(\int_{-\infty}^{\infty} S(t)\, \mathrm{e}^{-2\pi i\nu t}\, dt\right)^*\left(\int_{-\infty}^{\infty} S(\tilde{\tau})\, \mathrm{e}^{-2\pi i\nu\tilde{\tau}}\, d\tilde{\tau}\right) \\
&= |\mathcal{F}[S]|^2
\end{aligned} \tag{17}$$

In optics, the so called power spectrum describes the amount of power of the electromagnetic waves within a specific frequency range, also called "frequency bin". The power spectrum is defined as the absolute square of the Fourier transform of the electric field $E(t)$ which is exactly what we obtained in Equation (17) from the Fourier transform of the first-order autocorrelation function when the signal is the time-dependent electric field, i.e. $S(t) = E(t)$. Vice versa, the inverse Fourier transform of the power spectrum yields the first-order auto correlation function.

2.7 Photon Blockade and Resonance Fluorescence in Circuit QED

In Section 2.5 we have seen that the coupling of a transmon qubit to a coplanar waveguide resonator is described by the so called Jaynes-Cummings Hamiltonian that describes the interaction between an (artificial) atom and a specific resonator mode. Since the Jaynes-Cummings model originally developed to describe cavity QED experiments at optical

frequencies, it is natural to ask whether the effects implied by the Jaynes-Cummings model can also be observed in circuit QED. This question is particularly interesting because at the time of writing of the present thesis, techniques to generate and especially to detect single photons at microwave frequencies in superconducting circuits are under development with significant progress in the recent years [3–6, 18, 36]. In this section we explain a particular effect called photon blockade [37,38] that allows for the generation of a train of single photons. Furthermore we will see that electromagnetic radiation at the input of the cavity is absorbed and re-emitted incoherently by the atom-cavity system, an effect called resonance fluorescence.

In the context of the experiment described in Section 6 the question arises what the electromagnetic radiation at the output of the coplanar waveguide resonator looks like when coherent electromagnetic radiation approaches the input of the resonator.

For the following considerations, we approximate the coupled system formed when the qubit is in resonance with the cavity as an effective two-level system as already done for the qubit itself. This approximation is valid as long as the typical timescales of the dynamics we explore are much slower than the anharmonicity $(2-\sqrt{2})g$ of the dressed states.

If the coupled system is first in its ground state, then incident electromagnetic radiation with frequency $\omega_{1-} = \omega_\mathrm{r} - g$ causes the state of the coupled atom-cavity system to change from the ground state to the first excited state through the absorption of a photon. Unlike the case where the resonator is isolated from the qubit, the anharmonicity prevents the absorption of a second photon of frequency ω_{1-} since this frequency does not match any transition from the first excited state to higher excited states. This so called photon blockade forms the basis for the resonance fluorescence spectrum that is described in the rest of this section.

With the two-level approximation we write the effective Hamiltonian of the lowest two levels of the coupled system resonantly driven by the electromagnetic mode as [39]

$$\hat{H}_\mathrm{m} = \frac{1}{2}\hbar\omega_{1-}\hat{\sigma}_\mathrm{z} + \hbar\omega_{1-}\hat{b}^\dagger\hat{b} + \hbar\left(\tilde{\Omega}^*\hat{b}^\dagger\hat{\sigma}^- + \tilde{\Omega}\hat{b}\hat{\sigma}^+\right), \tag{18}$$

where $\hat{\sigma}_\mathrm{x}$, $\hat{\sigma}_\mathrm{y}$ and $\hat{\sigma}_\mathrm{z}$ are the Pauli matrices for the combined system of qubit and coplanar waveguide resonator. The annihilation operator $\hat{b}$ annihilates one excitation in the electromagnetic radiation mode of the drive. The coupling constant $\tilde{\Omega}$ is related to the dipole moment d and the electric field strength E through $\tilde{\Omega} = -dE/(\hbar\langle\hat{b}\rangle)$.

Diagonalizing the Hamiltonian given by Equation (18), it can be derived [39] that if the resonant drive is coherent and has a mean photon number much larger than 1, then driving the coupled atom-cavity system leads to another dressing of the dressed states $|n\pm\rangle$ of the Jaynes-Cummings model where the additional splitting depends on the Rabi frequency $\Omega_\mathrm{R} = 2dE/\hbar$ as depicted in Part (a) of Figure 5. This "supersplitting" was already observed in circuit QED [40].

To be able to accurately predict the power spectrum at the output of the coplanar waveguide resonator, one has to take into account the damping induced by the decay of the exited states of the resonator mode and the qubit which takes place at rates κ and γ respectively as already mentioned in Section 2.1. The damping of the combined system

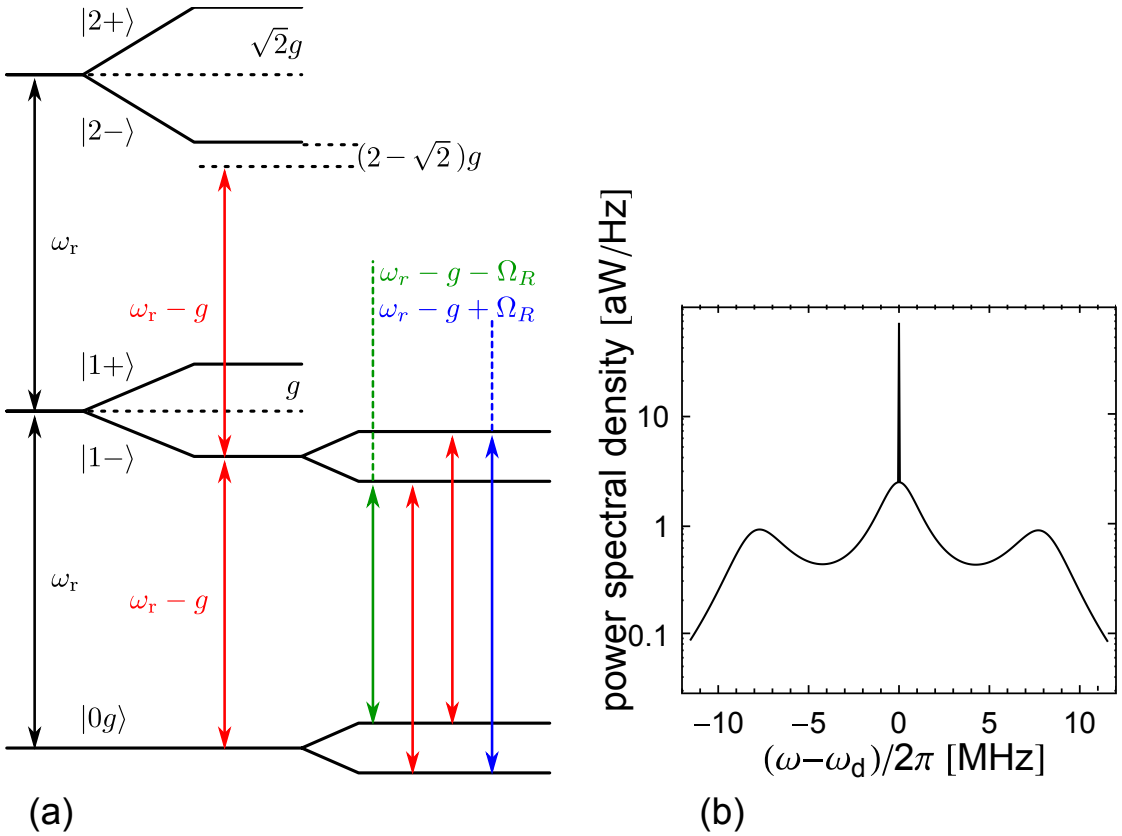


Figure 5: **(a)** Dressing of the dressed states of the Jaynes-Cummings model in the resonant case $\omega_\mathrm{a} - \omega_\mathrm{r} \ll g$. **(b)** Simulated spectrum of the resonance fluorescence of a resonator coupled to a qubit showing a Mollow triplet. The resonator has resonance frequency $\omega_\mathrm{r}/2\pi = 6.769\,\mathrm{GHz}$ and decay rate $\kappa/2\pi = 4\,\mathrm{MHz}$. The resonator is coupled to a qubit with a coherent dipole coupling strength of $g/2\pi = 73\,\mathrm{MHz}$. The qubit has a decay rate of $\gamma/2\pi = 0.4\,\mathrm{MHz}$. The drive amplitude corresponds to $\Omega_\mathrm{R}/2\pi = 7.9\,\mathrm{MHz}$. Figures reprinted from [6].

of resonator mode and qubit can be modeled as a coupling to an external radiation bath [39].

Mollow predicted the power spectrum of the electromagnetic radiation scattered by the damped two-level system [22]. The incident radiation is assumed to be a coherent drive on resonance with the transition frequency of the two-level system. Mollow calculated the first-order autocorrelation function of the electric field emitted by the two-level system. The predicted spectrum features a delta peak at the drive frequency which is thus identified as coherent scattering, referred to as Rayleigh scattering. When the drive amplitude corresponds to a Rabi frequency Ω_{R} much slower than the decay rate $\tilde{\kappa}$ of the two-level system, the coherent scattering dominates over resonance fluorescence.

On the other hand, if the two-level system is driven by a strong drive with $\Omega_{\mathrm{R}} \gg \tilde{\kappa}$, incoherent scattering becomes significant. The incoherent scattering leads to three different Lorentzian shaped peaks in the power spectrum. One of the incoherent peaks is predicted to be at the drive frequency where the coherent peak is located. In contrast to the Rayleigh scattering peak, this incoherent peak is much broader with width $\frac{1}{2}\tilde{\kappa}$ Furthermore, two satellite peaks originate from the dressing induced by the drive radiation as seen from Part (a) of Figure 5. The satellite peaks have a width of $\frac{3}{4}\tilde{\kappa}$. In the limit of strong driving, the separation in frequency domain of the satellite peaks to the coherent peak scales linearly with Ω_{R}. The three incoherent peaks arising from the resonance fluorescence of a two-level system are referred to as Mollow triplet.

3 Analog and Digital Signal Analysis

In the present thesis a measurement instrument is designed which can be used to digitally compute the power spectrum of a signal. In order to use this instrument for the measurement of e.g. the resonance fluorescence of a circuit QED system, the analog signal that is emitted from the circuit QED system has to converted into a digital signal. This is done using analog-to-digital converters. Analog-to-digital converters discretize the time-dependent voltage at their input. The discretization takes place both in time and voltage of the signal. In fact the analog-to-digital converter measures a set of tuples (t_n, x_n) of times t_n and discretized voltages x_n. The times t_n at which the voltage is measured are given by

$$t_n = t_0 + n/f_\mathrm{s} \qquad n = 0, 1, 2, 3, \ldots, N, \tag{19}$$

where t_0 is the starting time, f_s is the sampling frequency and N is the number of samples. The sampling frequency is defined by a sampling clock. The starting time t_0 is usually determined by a trigger. The number of samples N is usually limited by the available amount of digital memory.

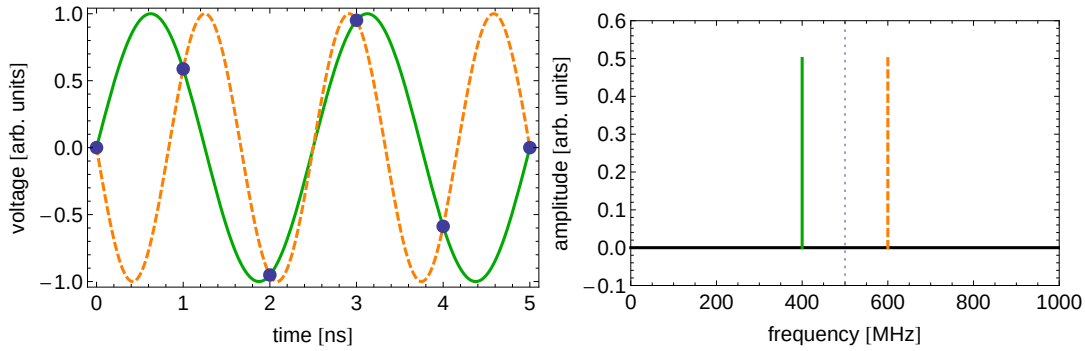
In an ideal analog-to-digital converter, the discretized voltages x_n depend on the actual voltages $V(t_n)$ as follows

$$x_n = \begin{cases} \mathrm{RoundToInt}\left(2^{B-1}V(t_n)/V_\mathrm{full}\right), & , -V_\mathrm{full} \leq V(t_n) \leq V_\mathrm{full}\left(1 - 2^{-(B-1)}\right) \\ 2^{B-1} - 1 & , V(t_n) > V_\mathrm{full}\left(1 - 2^{-(B-1)}\right) \\ -2^{B-1} & , V(t_n) < -V_\mathrm{full} \end{cases} \tag{20}$$

where "RoundToInt" denotes the function that maps its argument to the nearest integer, B is the number of bits of the digital words that represent the samples and V_full is the voltage that corresponds to the largest representable digital word.

A consequence of the time discretization with a sampling frequency f_s is the so called aliasing. Aliasing emerges because the waveform of the original analog signal can not be determined without ambiguity from the set of measured data points alone. For example, when the sampling frequency is $f_\mathrm{s} = 1\,\mathrm{GHz}$, the data points resulting from a 400 MHz signal can also be interpreted as coming from a corresponding 600 MHz signal and vice versa as can be seen from Figure 6. The 600 MHz signal has an "alias" at a frequency of 400 MHz, hence this effect is called "aliasing". In general, each real-valued signal with frequency f has aliases at the frequencies $f_N = |f - N f_\mathrm{s}|$ for all integers $N \in \mathbb{Z}$.

In the spectrum obtained from the sampled signal by the discrete Fourier transform, the aliasing has the effect that signal components with frequencies larger than $f_\mathrm{s}/2$ are folded back to the frequency bins which correspond to the aliased frequency components in the discrete spectrum. This ambiguity can be solved by the assertion that the analog signal only has components in a certain frequency band with a maximal width of $f_\mathrm{s}/2$. This can in principle be done by applying an analog band-pass filter to the input signal of the analog-to-digital converters. One disadvantage of the analog band-pass filters is that the center of the frequency band cannot easily be shifted. An alternative to band-pass filtering is provided by analog down conversion.



(a) two signals in time domain

(b) two signals in frequency domain

Figure 6: Example of two hypothetical signals yielding the same time-discretization. Part **(a)**, green solid line: sine wave with a frequency of 400 MHz, orange dashed line: sine wave with a frequency of 600 MHz. The blue dots represent the data points that are obtained when the signal is sampled with a frequency of $f_\mathrm{s} = 1\,\mathrm{GHz}$. Part **(b)** shows the two signals in the frequency spectrum. The dotted blue line at $f_\mathrm{s}/2 = 500\,\mathrm{MHz}$ marks half the sampling frequency.

Down conversion is a signal processing operation that shifts the spectrum of the signal by an amount corresponding to the down conversion frequency f_c towards lower frequencies. Analog down conversion is usually done with an IQ mixer. The IQ mixer is an electronic circuit which takes the radio frequency (RF) signal and a reference signal from a local oscillator (LO) as input. The LO signal is split into two signals and the phase of one of the two resulting signals is shifted by $\pi/2$. Each of these two signals is multiplied (mixed) with the RF signal. The result of these two multiplications is called in-phase (I) and quadrature (Q), hence the name IQ mixer. The effect of the IQ mixer on the signal can be interpreted as a complex multiplication. Let the LO signal be given by $S_\mathrm{LO}(t) = A_\mathrm{LO}\cos(\omega_\mathrm{LO}t)$, where A_LO is the amplitude and ω_LO the frequency of the LO signal. The effect of the IQ mixer on the input RF signal $S_\mathrm{RF}(t)$ is then given by the following equation [41]

$$\tilde{S}(t) = S_\mathrm{I}(t) + iS_\mathrm{Q}(t) = K\frac{A_\mathrm{LO}}{\sqrt{2}}\,\mathrm{e}^{-i\omega_\mathrm{LO}t}\,S_\mathrm{RF}(t), \tag{21}$$

where $S_\mathrm{I}(t)$ and $S_\mathrm{Q}(t)$ are the two outputs of the IQ mixer and K is a so called mixer constant which includes the losses due to reflections caused by the impedance mismatch induced by the mixer.

To see the effect of the analog down conversion by the IQ mixer on the spectrum of

the *complex* signal $\tilde{S}(t)$ the signal has to be Fourier transformed

$$\begin{aligned}
\tilde{F}(\omega) &= \int_{-\infty}^{\infty} \tilde{S}(t)\, \mathrm{e}^{-i\omega t}\, \mathrm{dt} \\
&= K\frac{A_{\mathrm{LO}}}{\sqrt{2}} \int_{-\infty}^{\infty} \mathrm{e}^{-i\omega_{\mathrm{LO}} t}\, S_{\mathrm{RF}}(t)\, \mathrm{e}^{-i\omega t}\, \mathrm{dt} \\
&= K\frac{A_{\mathrm{LO}}}{\sqrt{2}} \int_{-\infty}^{\infty} S_{\mathrm{RF}}(t)\, \mathrm{e}^{-i(\omega+\omega_{\mathrm{LO}})t}\, \mathrm{dt} \\
&= K\frac{A_{\mathrm{LO}}}{\sqrt{2}} F_{\mathrm{RF}}(\omega + \omega_{\mathrm{LO}}),
\end{aligned} \tag{22}$$

where $\tilde{F}(\omega)$ is the Fourier transform of $\tilde{S}(t)$ and $F_{\mathrm{RF}}(\omega)$ is the Fourier transform of the RF signal $S_{\mathrm{RF}}(t)$. From this it can be seen that the IQ mixer shifts the spectrum of $\tilde{S}(t)$ by $-\omega_{\mathrm{LO}}$.

Since the complex signal $\tilde{S}(t)$ consists of the two real signals $S_{\mathrm{I}}(t)$ and $S_{\mathrm{Q}}(t)$ it is also important to know the effect of the IQ mixer on the spectrum of each of these real signals. The Fourier transform of $S_{\mathrm{I}}(t)$

$$\begin{aligned}
F_{\mathrm{I}}(\omega) &= \int_{-\infty}^{\infty} S_{\mathrm{I}}(t)\, \mathrm{e}^{-i\omega t}\, \mathrm{dt} \\
&= K\frac{A_{\mathrm{LO}}}{\sqrt{2}} \int_{-\infty}^{\infty} \cos(\omega_{\mathrm{LO}} t) S_{\mathrm{RF}}(t)\, \mathrm{e}^{-i\omega t}\, \mathrm{dt} \\
&= K\frac{A_{\mathrm{LO}}}{\sqrt{2}} \int_{-\infty}^{\infty} \frac{\mathrm{e}^{i\omega_{\mathrm{LO}} t} + \mathrm{e}^{-i\omega_{\mathrm{LO}} t}}{2} S_{\mathrm{RF}}(t)\, \mathrm{e}^{-i\omega t}\, \mathrm{dt} \\
&= K\frac{A_{\mathrm{LO}}}{2\sqrt{2}} \left(\int_{-\infty}^{\infty} S_{\mathrm{RF}}(t)\, \mathrm{e}^{-i(\omega+\omega_{\mathrm{LO}})t}\, \mathrm{dt} + \int_{-\infty}^{\infty} S_{\mathrm{RF}}(t)\, \mathrm{e}^{-i(\omega-\omega_{\mathrm{LO}})t}\, \mathrm{dt} \right) \\
&= K\frac{A_{\mathrm{LO}}}{2\sqrt{2}} \left(F_{\mathrm{RF}}(\omega + \omega_{\mathrm{LO}}) + F_{\mathrm{RF}}(\omega - \omega_{\mathrm{LO}}) \right).
\end{aligned} \tag{23}$$

Thus for each frequency component of the input signal $S_{\mathrm{RF}}(t)$, we obtain two copies. One copy is at a frequency of $\omega - \omega_{\mathrm{LO}}$ and the other at a frequency of $\omega + \omega_{\mathrm{LO}}$. The Fourier transform of the signal $S_{\mathrm{Q}}(t)$ can be computed accordingly.

In order to prevent aliasing in the analog-to-digital conversion, analog down conversion is used to shift the relevant spectral components of the signal at the output of the resonator from microwave frequencies into the regime which is below $f_{\mathrm{s}}/2$.

The frequency at which the main component (carrier) of the resulting signal is expected after the analog down conversion is called intermediate frequency (IF). Because of the

analog down conversion, the IF is related to the original carrier frequency by $\omega_{\mathrm{IF}} = \omega_{\mathrm{carrier}} - \omega_{\mathrm{LO}}$. The frequency components which are higher than $f_{\mathrm{s}}/2$ are attenuated by an analog low-pass filter after the analog down conversion. In this manner, analog down conversion followed by low-pass filtering before analog-to-digital conversion has a similar effect as a tunable band-pass filter preceding the analog-to-digital conversion would have.

The digitized signal can be analyzed by methods of digital signal processing as described in Section 4.7.

4 FPGA-Based Measurement Instrument

4.1 Background on FPGA Programming

The intent of this chapter is to familiarize the reader with the concepts and terminology of field programmable gate array (FPGA) programming relevant to understand the description of the design of an FPGA-based measurement instrument that follows in Sections 4.2 to 4.10. We begin with a motivation for why FPGAs are a good choice for digital signal processing in Section 4.1.1. Then we continue with a description of the concept and components of FPGAs in Section 4.1.2 and conclude with a small overview of the tools needed for FPGA programming in Section 4.1.3.

4.1.1 Introduction

For applications in signal processing and automation in which a large amount of data has to be processed at very high speeds or feedback control has to be realized, the standard CPUs built into todays computers are usually not fast enough. The major drawback of these processors is that they are programmed by a set of instructions which are processed one by one in a sequential manner. To speed up the computation with CPUs one has to either increase the rate at which the instructions are processed or have many CPU cores that process the instructions in parallel. The former alternative does not work when data has to be processed at rates close to the limit of the frequency at which the processors can be clocked. The latter possibility takes along some major difficulties when it comes to distribute slices of the data to the different processing units and when the computation on one processor depends on some intermediate results obtained by another processor.

The traditional way to overcome the above stated problems is to design specialized integrated circuits. These so called application specific integrated circuits (ASIC) cannot be programmed but have a fixed structure which maps the input signal to the output signal in the desired way. The fact that the desired algorithm is directly implemented in hardware makes it possible to have many parallel processing units that perform a single operation on a very small area and thus minimize the delay which is introduced by routing the data to the corresponding processing units.

Another very important aspect of having an ASIC is the possibility to pipeline the entire processing. Pipelining is parallelism where successive operations of an algorithm are performed all at the same time on different sets of data. This approach is very natural and has its close analogy for example in the production of food where the farmer may harvest crop at the same time as the already harvested corn gets milled and bread is baked from flour that the mill already produced.

The ASIC have the major drawback that once they are produced, the algorithm they perform cannot be changed anymore. The idea behind so called field programmable gate arrays (FPGA) is to build a generic circuit in which the interconnects can be switched on and off and the operation of each block in the circuit can be configured. An FPGA is an integrated circuit on which an arbitrary digital circuit can be programmed without making changes to the underlying semiconductor.

4.1.2 Primitives

In this section the basic and advanced primitives of an FPGA are described. This is done with a focus on the aspects of the Xilinx Virtex-6 architecture which are relevant for the present thesis.

The programmability of the FPGAs mentioned in Section 4.1.1 is achieved by two main ingredients (primitives): the so called lookup tables (LUTs) and programmable switch matrices (PSM) that interconnect the LUTs.

The LUTs are random access memory elements in which the truth table of a logical operation is stored. The LUTs thus implement a configurable logic function by interpreting the input as the memory address at which the desired output bit is stored. A LUT with n binary inputs and 1 output stores 2^n output bits. In the Xilinx Virtex-6 architecture, the LUTs have 6 inputs and 2 independent outputs [42]. In the case of the Virtex-6, a group of 4 LUTs comprise a slice which also contains 8 storage elements to optionally register or latch the outputs of the LUTs.

In the Xilinx terminology two adjacent slices together are called a configurable logic block (CLB). As depicted in Figure 7, each CLB is connected to a programmable switch matrix. The PSM allows to route the outputs of one CLB to the input of another CLB by establishing switchable connections between horizontal and vertical wires [43]. Figure 7 does not accurately describe the Virtex-6 architecture since the number of horizontal and vertical lines is larger in the Virtex-6.

The PSM offers different levels of connectivity: as can be seen from Figure 7 some wires connect only to every second CLB, the so called double lines. One double route introduces less routing delay than two singe lines because the switches introduce a delay in addition to the propagation time of the signal. In the Virtex-6 there are single, double, and quad lines which connect to every single, every second, and every fourth CLB respectively [44].

In addition to the switching matrix, the slices in the same column are connected by direct links which are intended to communicate the carry bit when the slices are used as full adders. Not shown in Figure 7 is the global clock tree which consists of lines that directly connect from the so called clock management tiles (CMT) to every CLB such that the skew and jitter is minimized and to save the resources of the programmable switch matrices for data transfer. Each line of the global clock tree consists of two wires. The voltage difference between these two wires encodes the actual clock signal. This concept, called differential signaling, reduces the sensibility of the signal transmission to noise in the supply voltage. The opposite of differential signaling is called single-ended signaling for which the voltage between one wire and the ground encodes the digital signal.

In the Xilinx Virtex-6 architecture, there exist additional, specialized primitives such as input/output blocks (IOB), dedicated slices for digital signal processing (DSP slices) and the block random access memory (block RAM). In the following, each of these special primitives are described.

Each IOB manages the input and output of signals The IOB contain e.g. input buffers to map low-voltage differential signals (LVDS) at pairs of input pins onto the internal

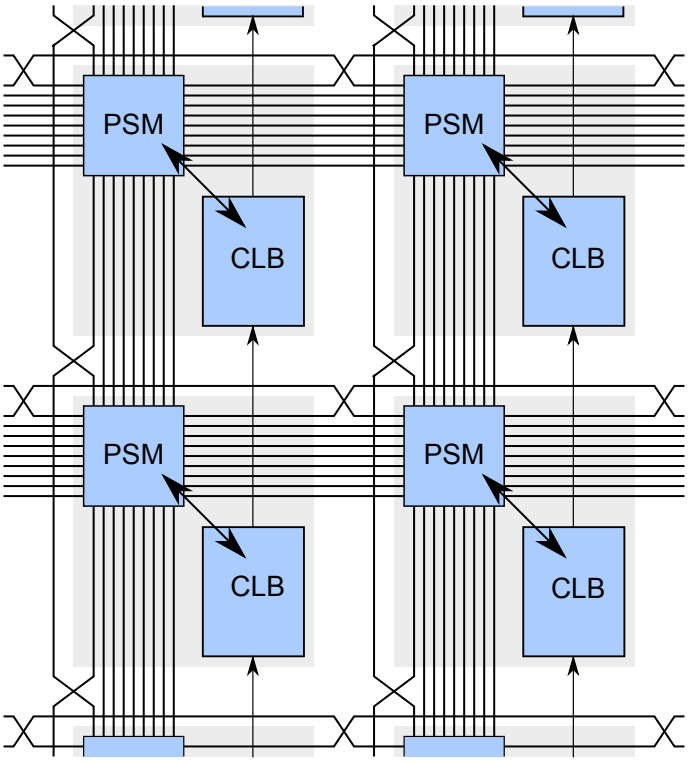


Figure 7: Schematic drawing of the basic elements of an FPGA showing some aspects of signal routing. Figure adapted from [43].

single-ended signal routes of the FPGA. The IOB also contain the deserializer and input delay described in Section 4.6.1.

The DSP slices in the Virtex-6, called DSP48E1 [45], are designed for the implementation of digital signal processing such as the fast Fourier transform described in Section 4.7.2. Each DSP48E1 slice contains a two-input multiplier followed by a three-input adder/subtracter which can be configured to act as an accumulator. Each of these elements can be bypassed if it is not needed. The multiplier takes two input arguments in the two's-complement format, one input having a width of 25 bit and the other input having a width of 18 bits. The accumulator which contains the adder/subtracter is 48 bits wide and can also be used as two or four independent parallel adders/subtracters with a width of 24 bits or 12 bits respectively.

On the Virtex-6, there are slices that can either be used as memory elements, so called distributed RAM, or as LUT to implement logic operations. These slices are called SLICEM as opposed to the so called SLICEL which cannot be used to implement distributed RAM. Each of the 4 LUTs in the SLICEM can be selected to contribute to the distributed memory or not. Each CLB can either contain 0 or 1 SLICEM. Every CLB which contains a SLICEM can contribute a maximum of 256 bits of memory to the distributed RAM by using all 4 LUTs of the SLICEM. In total there are 14600 SLICEM primitives on the VLX240T type of Virtex-6 FPGA which limits the total amount of distributed RAM to about 3.7 Mbit of RAM with variable width. However, one has to take into account that a relatively large amount of slice LUTs have to be used for the implementation of logic gates. E.g. the firmware described in Section 4.3 uses about 27% of all available LUTs for logic operations. Another problem is the signal routing, i.e. the problem of placing and connecting the LUTs in such way that for example the clock frequency the designer needs can be realized without data loss as described in Section 4.4.1.

The Virtex-6 architecture also offers an alternative to distributed RAM which is called block RAM. Block RAM is composed of specialized primitives that are arranged in separate columns of the FPGA layout. The block RAM offers dual port capabilities which means that there are two distinct interfaces that can be used to write and read the memory at the same time as long as address collisions are avoided [46]. Note that in the case of both operations being read operations, address collisions are allowed.

There are two distinct dual port modes: the "true" dual port mode where the RAM can be read/written simultaneously over two independent interfaces and the "simple" dual port mode where one interface is read-only whereas the other is write-only. The size of the Virtex-6 block RAM primitives (RAMB36E1) is 36 Kbit per block in the true dual port mode or 72 Kbit in the simple dual port mode. In the VLX240T type of Virtex-6 FPGA there is a total of 416 RAMB36E1 primitives which implies a maximum of 14976 Kbit of true dual port RAM. Each RAMB36E1 can also be used as two independent 18 Kbit (RAMB18E1) blocks. Refer to [46] for all feasible combinations of width and depth of the block RAM primitives.

4.1.3 Software Tools

The ability to configure each primitive of an FPGA implies a large number of degrees of freedom as there are for example 150720 LUTs in the Virtex-6 VLX240T FPGA manufactured by Xilinx, Inc. [47] each of them having $2 \times 2^6 = 128$ entries. Thus it is very important to be able to describe the design of the virtual circuit to be realized on the FPGA at a higher level of abstraction. For this purpose two main standardized hardware description languages (HDL) were established in the engineering world: the very high speed integrated circuit (VHSIC) hardware description language (VHDL) [43] and Verilog [48]. Both of these languages allow the designer to split the design into modules which can be nested and instantiated multiple times. Furthermore, these languages can be used to describe clock synchronous and asynchronous processes to be realized in the logic circuit by means of conditional and iteration statements such as the classical "if-then-else" and "for-loop" statements which are well known from sequential programming languages such as C. The main difference between HDLs and sequential programming languages is that in HDLs all of the commands are "executed" in parallel. For example an element-wise vector addition programmed using a VHDL for loop will be compiled to a low level description of the hardware design in which the elements of the two vectors will not be added one by one but instead the circuit will consist of as many digital adders as there are elements in the vector which all operate independently and thus in parallel.

Compilers for hardware description languages such as the "ISE Design Suite" by Xilinx, Inc. which has been employed for the present thesis, usually consist of a collection of tools for each stage of the FPGA design flow. The design flow typically consists of the following basic stages: synthesis, mapping, placement and routing. In the synthesis stage a data structure, sometimes called "netlist", that lists which primitive components such as logic gates and flip-flops are instantiated and how they are connected to each other, is compiled from the HDL description of the design. The netlist produced in the synthesis stage is usually independent of the type of FPGA on which it is to be realized. Therefore the primitives from the netlist have to be mapped to the resources that exist on the chip layout of the FPGA (i.e. LUTs, flip-flops, and some specialized blocks) in the mapping stage. As there are many LUTs, there are many possibilities to choose one particular LUT for the implementation of one particular logic gate of the netlist. In the placement and routing stage, the optimal placement and interconnect of the primitives in the matrix of LUTs and PSM is determined such that all timing constraints imposed by the designer are met. For more information about timing constraints refer to Section 4.4.1.

Since finding the optimal placement and interconnections of the primitives of an FPGA design is a high dimensional optimization problem, the tools that exist to perform the placement and routing task automatically, rely on a heuristic optimization algorithm and thus do not always find the global optimum for the FPGA configuration and sometimes even fail to find a local optimum that meets the timing constraints.

The result of every stage of the FPGA design flow can be used to obtain a simulation model by simulation software such as iSim which is part of the Xilinx ISE design suite or the more wide-spread ModelSim by Mentor Graphics. The simulation model describes

how the input and output signals of each primitive depend on time. Note that at a high level of abstraction, e.g. before synthesis, it is not possible to obtain an accurate prediction of the timing as there is no information on how the design will be implemented in real hardware.

More recently there has been an effort in the FPGA industry to produce graphical programming languages that are primarily intended for the modeling of digital signal processing (DSP) systems and thus operate at an even higher level of abstraction than HDL. Examples for such tools are the "System Generator for DSP" [49] which is also part of the Xilinx ISE Design Suite and the "DSP Builder" by Altera Corporation which is the main competitor of Xilinx. Both, the "System Generator for DSP" as well as the "DSP Builder" depend on the Simulink design environment by MathWorks, Inc which is a graphical platform for the simulation and model-based design of dynamical systems. In Section 4.7, we present how these tools are applied for the design of the digital signal processing system that has been developed within the present thesis.

4.2 Hardware Overview

In this section we present the hardware we have selected to achieve the goal of the present thesis: performing correlation function measurements at a sampling frequency of 1 GHz while averaging over a large number of measurement shots and providing the basis for future applications which need high-bandwidth measurements. The latter purpose includes the requirement for high flexibility. Thus we were looking for a solution which offers enough analog input and output channels at the analog front end in addition to the basic requirements such as to be able to synchronize to external trigger sources, e.g. arbitrary waveform generators. High flexibility is also required for the ability to process the digitized signal in different ways at high data throughput rates. In Section 4.1 it is emphasized that field programmable gate arrays offer the advantages of application specific circuits together with the ability to reconfigure the circuit. There exist commercial off-the-shelf products which offer print circuit boards which comprise the FPGA as well as various peripheral components such as analog-to-digital converters, storage solutions and interfaces to other computational units such as other FPGA or personal computers.

The group in which this thesis is written has already made productive experience with FPGA-based measurements [4, 6, 50–52]. The system used in the previous experiments is based on the "XtremeDSP Development Kit IV" which is manufactured by Nallatech Ltd. and uses the Virtex-4 FPGA by Xilinx [53]. It is beneficial to look for a solution that has the most features in common with the existing solution in order to be able to reuse as much as possible of the existing firmware and interface design. Unfortunately, there is no successor of the XtremeDSP Development Kit IV which offers the required minimal sampling rate of one billion samples per second (1 GSPS) at the time of writing this thesis.

In the time since the previous FPGA-based measurement system has been designed, a new standardized interface has been approved by the American National Standards Institute (ANSI), namely the VITA 57 FPGA Mezzanine Card (FMC), which is intended to connect peripheral devices by means of extension cards to FPGA on print circuit

boards [54]. This allows to have an FPGA mounted on a motherboard and to interchange e.g. the analog-to-digital converter by connecting different FMC boards produced by different manufacturers in the same manner as todays personal computers offer the PCI connectors to connect extension cards. This concept fits well to the requirement for high flexibility for future adjustments.

As motherboard, we have chosen the Xilinx ML605 development board [55] which has the Xilinx Virtex-6 XC6VLX240T-1FFG1156 FPGA as its heart as shown on Figure 8. In the following, it is described how to interpret the model number. The first part of the model number (XC6VLX240T) identifies which type of Virtex-6 FPGA it is. The types differ in the number of available primitives such as LUTs, DSP Slices, Block RAM etc. The Virtex-6 Family Overview [47] gives an overview over the available types. The -1 after the type identifier stands for the so called speed grade. The speed grade determines among other things the maximal clock rate with which the primitives such as the DSP slices can be clocked [56]. The -1 speed grade is the lowest speed grade. Finally, the FFG1156 stands for the packaging. The package determines the number of pins and their arrangement. Different type of the Xilinx Virtex-6 FPGA with the same package exist which are thus interchangeable without the need of a new print circuit board design.

As seen from Figure 8, the board offers two FMC connectors: one of them accords to the low pin count (LPC) standard and the other to the high pin count (HPC) standard. For temporary data storage, the board offers a double data rate (DDR3) small outline dual in-line memory module (SODIMM) slot filled with 512 MB of random access memory (RAM). The board is intended to demonstrate as many features of the Virtex-6 as possible and thus offers great flexibility through many interfaces such as e.g. the tri-mode 10/100/1000 Mbit/s Ethernet interface and the 8-lane PCI Express interface which can be used to connect the FPGA to a host computer. The PCI Express interface allows for transfer rates of 2 GB/s by either operating in the 4-lane, generation 2 mode with a transfer rate of 5 giga-transfers per second or in the 8-lane, generation 1 mode with a transfer rate of 2.5 giga-transfers per second.

For the analog front end we have chosen the FMC110 mezzanine card [57] shown on Figure 9 which is manufactured by 4DSP LLC and is compatible with the FMC high pin count (HPC) standard. The FMC110 comprises two analog-to-digital converters (ADC0 and ADC1) and two digital-to-analog converters (DAC0 and DAC1) clocked by a clock distribution chip that implements a phase-locked loop which can be locked to an external clock through a dedicated clock input connector at the front panel. The two ADS5400 analog-to-digital converter microchips on the FMC110 card are manufactured by Texas Instruments and can be operated at a maximum rate of one giga-samples per second (1 GSPS) at a resolution of 12 bits per sample [58]. Also manufactured by Texas Instruments are the digital-to-analog converter DAC5681Z microchips which also operate at a maximum rate of 1 GSPS but at a higher vertical resolution of 16 bits per sample [59].

Furthermore, the FMC110 card includes a separate pin for the trigger input which is DC coupled to a low voltage differential signal (LVDS) channel that connects through the FMC connector to the FPGA. In addition there is a high definition multimedia interface (HDMI) connected to the FPGA through the FMC port via a low voltage transistor-

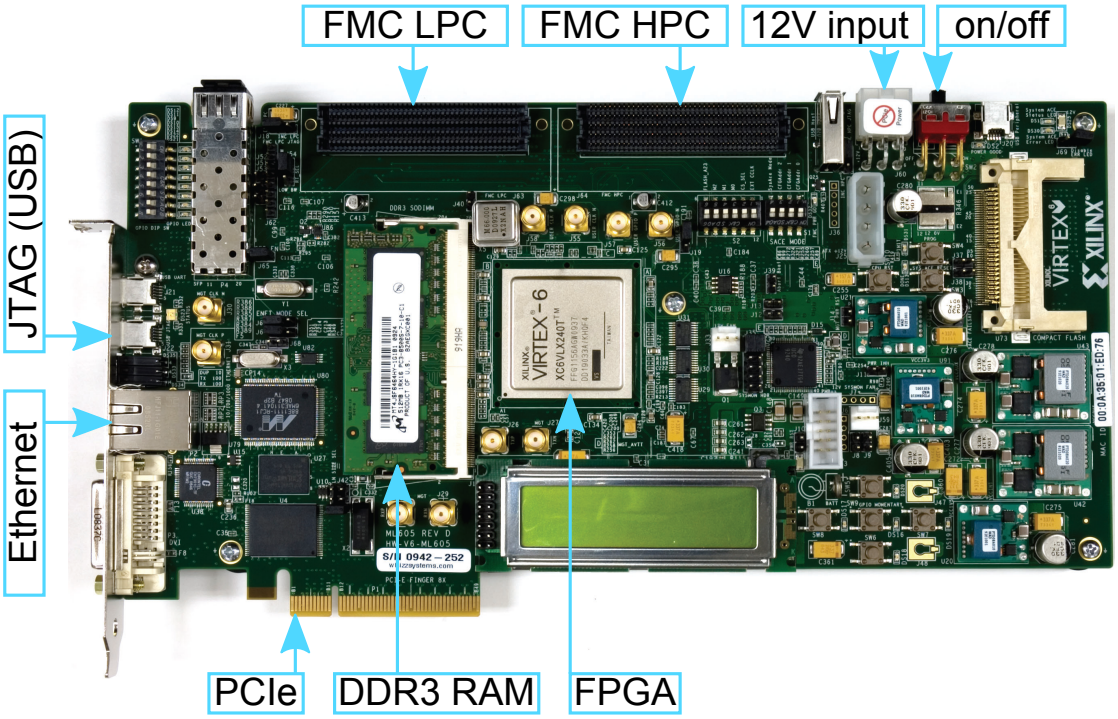


Figure 8: Xilinx ML605 development board with labeling of the most important parts. The photograph was published by Xilinx, Inc. on page 11 of the ML605 hardware user guide [55].

transistor logic (LVTTL) channel. The HDMI interface can be used for additional digital input and output such as additional trigger channels.

The FMC110 also features a 1 GHz voltage controlled crystal oscillator (VCXO) which is used in a phase-locked loop implemented by a AD9517-3 clock tree chip from Analog Devices, Inc. [60]. The phase-locked loop is used to derive the sampling clock of the analog-to-digital converter as well as to send a reference clock signal to the FPGA.

The phase-locked loop implemented by the AD9517-3 can take the reference clock signal from the clock input pin of the FMC110 card or alternatively from an on-board 100 MHz crystal (XTAL) oscillator. Refer to Section 4.5 for details on the initialization of the different components of the FMC110 mezzanine card.

For monitoring purposes, the FMC110 card has an ADT7411 temperature sensor and analog-to-digital converter by Analog Devices, Inc. [61] with which the temperature and different voltages on the card can be measured as described in Section 4.10.

The ADT7411 is made accessible to the FPGA by a I^2C bus which is connected to the FPGA through the FMC interface. The FMC standard stipulates the presence of an electrically erasable programmable read-only memory (EEPROM) on the mezzanine card which can be used to store an identifier of the card or to store error reports in a non-volatile way. The EEPROM which is mounted on the FMC110 is also attached to the I^2C bus.

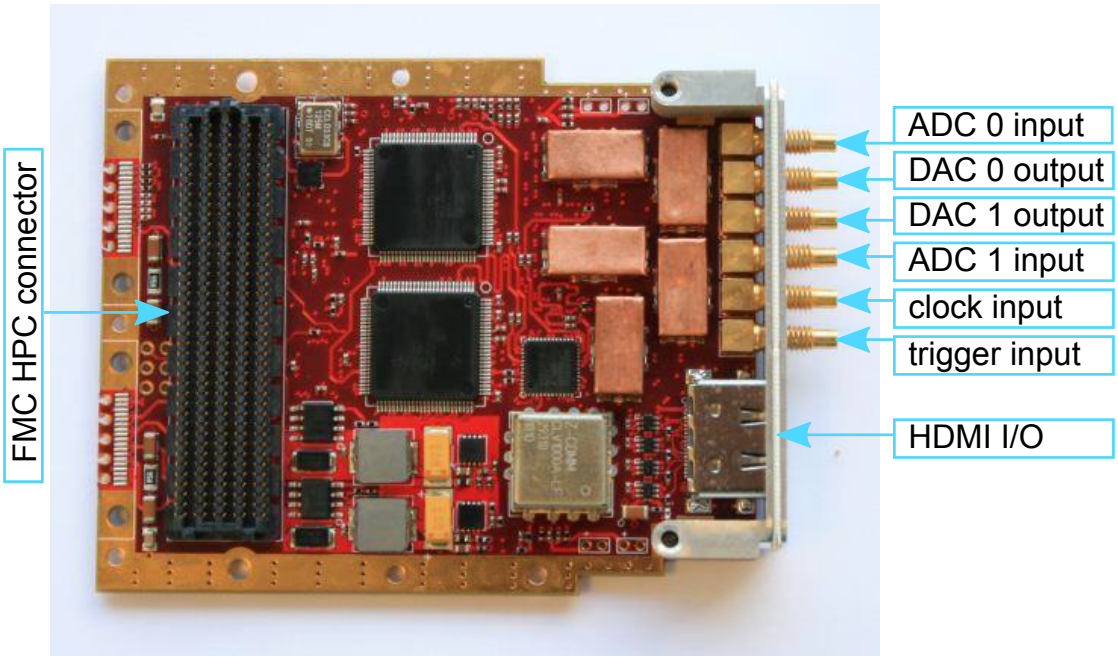


Figure 9: FMC110 mezzanine card manufactured by 4DSP LLC. All interfaces are labeled. The connectors on the front panel (right side of the card) except the HDMI plug confirm to the SSMC coaxial connector standard. The photograph was published by 4DSP LLC in the FMC110 user manual [57].

4.3 Firmware Concept

The central unit which connects the different hardware entities presented in Section 4.2 is formed by the FPGA and in particular by the circuit which is realized by the configuration of the FPGA. This virtual circuit is referred to as "firmware". Due to its central position, the firmware has to connect and control the different entities such as the analog-to-digital converter, external memory and Ethernet interface. In this section we present the concept of the firmware we have designed to perform correlation function measurements at a sampling frequency of 1 GHz.

Figure 10 gives an overview of the modules that are important in the data flow of measurement and signal processing. The top-level module, called "wrapper_virtex6", instantiates and interconnects all submodules of the design. The module "ml605_fmc110" contains those modules that are integrated into the so called "Stellar IP" framework which originates from the reference design for the FMC110 by 4DSP LLC [62]. The Stellar IP framework specifies interfaces to connect and address the individual modules of the design. Modules within the Stellar IP framework have the prefix "sip_".

In the following, the concept of the firmware is presented by explaining the data flow in a typical measurement and by listing the requirements for each step in the signal processing.

First, the time-dependent voltage of the analog signal has to be digitized. For this to be achieved, the components on the FMC110 mezzanine card have to be set up correctly. This is done via the interface logic between FPGA and FMC110 card implemented in VHDL in the module called "sip_fmc110". The digitized samples are then sent one by one in a serial manner directly from the analog-to-digital converter to the FPGA at a rate of one billion samples per second. The synchronization of this data transfer is also taken care of by the logic in the sip_fmc110 module as described in Section 4.6.1.

The second stage is the digital signal processing which is described in Section 4.7. The module in which the digital signal processing is performed has to be easily interchangeable in order to allow for different kinds of measurements which may require very different kinds of signal processing that would not fit into the available resources of the FPGA all at the same time. Thus it needs a generic interface to which e.g. the Ph.D. student who wants to perform a special kind of measurement can hook his/her application. For this reason we call the DSP module "sip_user_app_component".

During and after digital signal processing, data has to be stored. As explained in Section 4.1.2, there is about 1.8 MB of dual port block RAM on the Virtex-6 VLX240T. Because this amount is somewhat limited and because timing constraints are harder to reach the more FPGA resources are allocated, it might be necessary to swap data to the external RAM. Thus the firmware includes the "sip_mem_if" module which can write and read data to and from the external RAM through the memory controller implemented by Xilinx in the "u_memc_ui_top" module as described in Section 4.8.

Finally, data needs to be transfered to the host computer for post-processing and storage. This is done by sending the data over the Ethernet media access control (MAC) interface which is implemented by the module called "sip_mac_engine" programmed by 4DSP. The "sip_mac_engine" module also allows the host computer to send commands

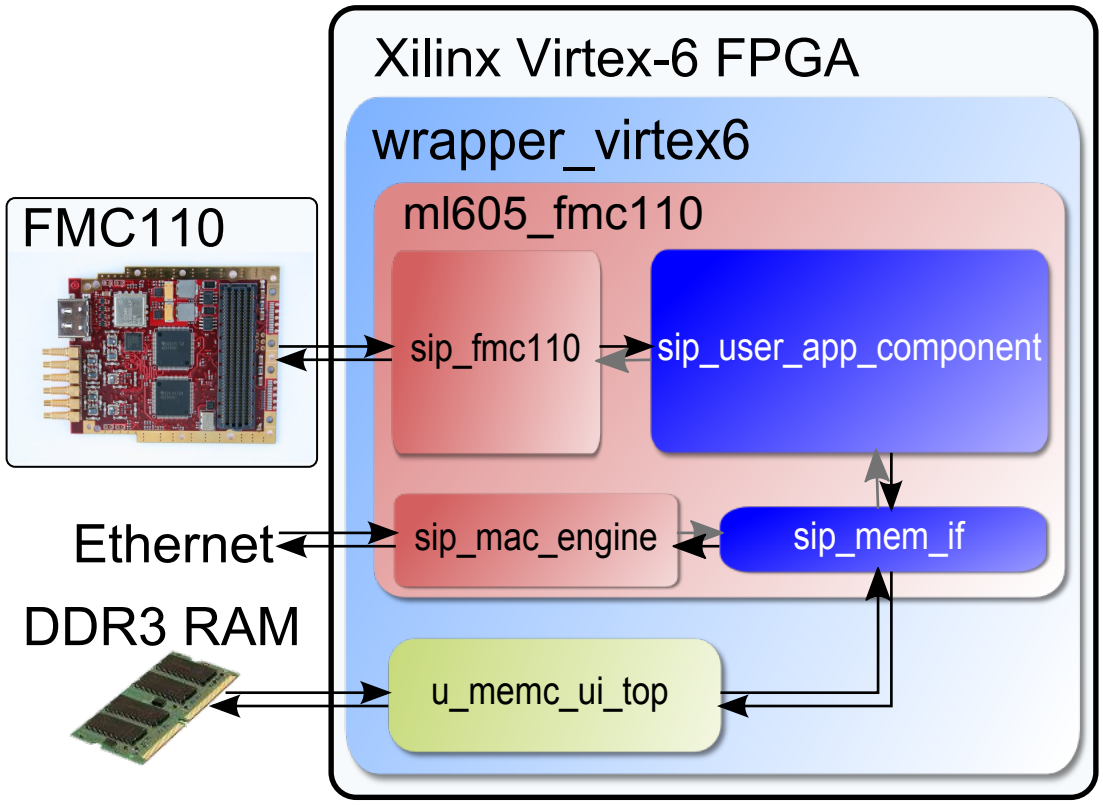


Figure 10: Overview of the FPGA design for measurement with digital signal processing. The boxes represent modules with the names of the corresponding VHDL implementation written inside the box. Only the most important modules are shown. The modules shaded in blue were implemented by the author of the present thesis whereas the red-colored boxes indicate modules which were originally implemented by 4DSP LLC. The green-colored module "u_memc_ui_top" was implemented by Xilinx, Inc. The arrows indicate interfaces across which data can be sent from one module to the other.

to each of the modules within the Stellar IP framework.

An attractive alternative to the Ethernet interface would be to use the PCI Express interface since it allows for faster transfer rates as stated in Section 4.2. This would, however, require the implementation of an interface driver on both the host side as well as on the FPGA which was not possible within the time frame of the present thesis. A Xilinx application note by Wiltgen and Ayer [63] provides a possible basis for such a driver.

4.4 Clocking

The frequency requirements of the firmware modules presented in Section 4.3 are not compatible for all modules. Therefore the firmware has to be divided into different clock regions. The present section is organized as follows: after an introduction of the basic limitations of FPGA clocking, some general considerations to manage data transfer across clock domains are presented. After that, we describe the different clock domains in the firmware that has been developed in the present thesis.

4.4.1 Frequency Limitations

In an FPGA design there are synchronous and asynchronous operations. Asynchronous operations do not need a clock signal since they immediately react on changes of their input signal. In contrast, synchronous operations, e.g. flip-flops, only change their internal state as well as their output when the dedicated digital clock input changes from 0 to 1 (rising edge) or from 1 to 0 (falling edge). These events are called clock events.

In the usual case where the synchronous elements are flip-flops, the parameters which determine the maximal frequency at which the synchronous elements can be clocked are the flip-flop clock-to-output delay T_{cko}, the routing time T_{route}, the combinatorial delay T_{logic} and the so called setup time T_{setup}. The clock-to-output delay T_{cko} is the time difference between the time the flip-flop received a clock event and the time when the output of the flip-flop has been updated. The parameter T_{route} is the time needed to propagate a signal change through all asynchronous operations in the path between two flip-flops and T_{logic} is the processing time of the asynchronous operations. Finally, the setup time T_{setup} is the minimal amount of time the signal has to be held constant at the input of the next flip-flop *before* the arrival of the clock event which causes the flip-flop to adjust its state to the input signal. The clock period T_{clk} has to be larger than the sum of all of these parameters [64]

$$T_{\text{clk}} \geq T_{\text{cko}} + T_{\text{route}} + T_{\text{logic}} + T_{\text{setup}}. \tag{24}$$

Note that in general also the so called hold time T_{hold} is relevant. The hold time is the amount of time during which the signal has to be held constant *after* the clock event. But since the hold time requirement in Xilinx FPGA is usually 0 [64], T_{hold} does not have to be taken into account in Equation (24).

4.4.2 Clock Domain Crossing

For the signal processing within a particular clock domain, the compilers in the mapping, placement and routing stage which have been introduced in Section 4.1 check whether the timing constraint given by Equation (24) is fulfilled. In contrast to signal processing within a single clock domain, special care has to be taken when exchanging digital signals between different clock domains. The phase difference between the clock signal of the two clock domains has to be taken into account. If the phase and frequencies of the two clocks are not locked to each other, the phase difference is random so there is no way to reliably ascertain that the signal is held constant during the setup time T_{setup} of the target flip-flop if the signal from the source changes in every clock cycle of the target clock domain. The signal has to be held constant for a long enough time until the flip-flop of the target clock domain has adapted to the change. An additional problem arises when the voltage of the output of the source is changing at the point in time at which the target flip-flop is clocked. In this case the target flip-flop will become metastable, i.e. its output will oscillate between the 0 and 1 levels until it recovers from the metastable state [65–67].

A way to transfer a pulse from one clock domain to another is implemented by 4DSP in the module "pulse2pulse" of the FMC110 reference design. This module implements a handshaking method which comes at the expense of a latency of at least 3 clock cycles in the target clock domain and assumes that no additional pulses occur in the source clock domain during the handshaking. Xilinx provides a first in first out (FIFO) memory module that supports input and output data synchronous to different clock domains. Both of these method need the clock signals of both clock domains as an input.

4.4.3 Clock Domains in the Firmware

In this section it is described how the different clock domains of the firmware arise and which relations exist between them. This description is accompanied by Figure 11 which shows the division of the modules of the firmware into parts that are clocked at different frequencies and/or from different clock sources.

The frequency of the clock domain which is used for the digital signal processing in the firmware developed in the present thesis is defined by the interface to the FMC110 card by 4DSP. It is necessary to process the data at a lower frequency than 450 MHz since the DSP-dedicated resources of the Virtex-6 can only be operated at a maximum frequency of 450 MHz [56]. Therefore the FMC110 interface module "sip_fmc110" is clocked at a frequency of 125 MHZ which equals to 1/8 of the sampling frequency $f_{\text{s}} = 1\,\text{GHz}$. Data from and to the FMC110 is received and sent in parallel as described in Section 4.6.1. The clock source for this clock domain is provided by a phase-locked loop implemented by the so called MultiMode Clock Manager (MMCM) primitive of the Xilinx Virtex-6. The MMCM for the interface to the analog front-end is locked to a differential clock signal that is sent from a dedicated clock chip on the FMC110 to the FPGA. A problem that arises with this design is that the clock chip on the FMC110 first has to be initialized by the FPGA. Therefore another clock domain is needed that runs synchronous to a

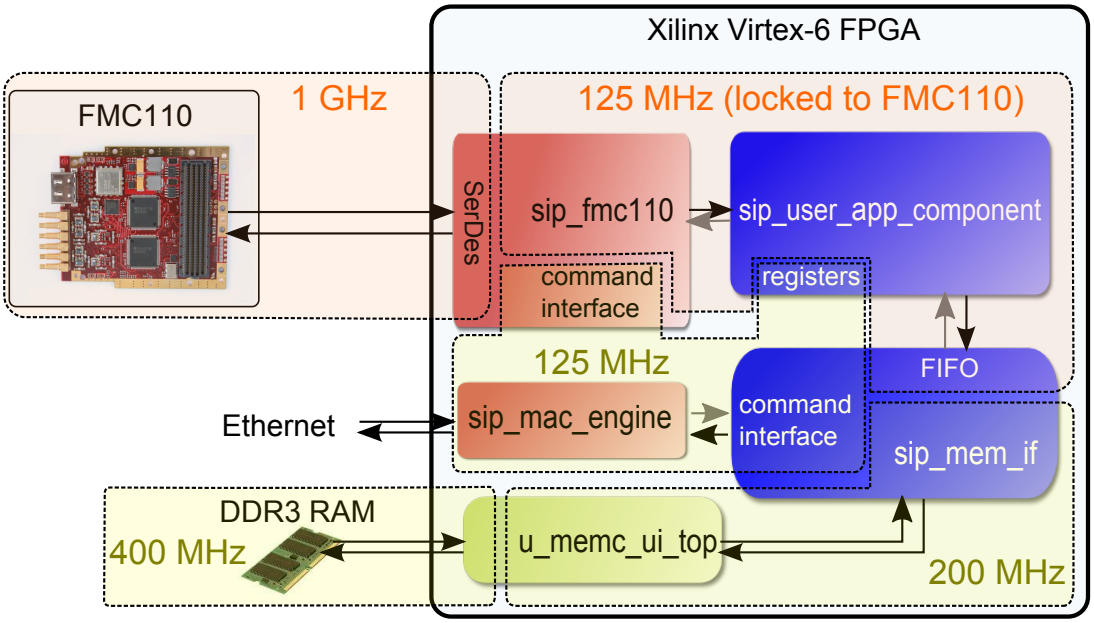


Figure 11: Division of the firmware modules into different clock domains. The boxes with dashed lines represent the different clock domains. Those clock domains which are indicated by orange boxes, i.e. the 1 GHz clock domain and the 125 MHz clock domain that has the words "(locked to FMC110)" in its label are locked to the phase-locked loop on the FMC110 mezzanine card. The other clock domains are locked to a 200 MHz oscillator on the Xilinx ML605 motherboard which is indicated by yellow boxes.

phase-locked loop which is locked to a permanently running 200 MHz oscillator on the ML605 board.

The memory interface module "u_memc_ui_top" operates at half the rate of the actual memory interface clock. Xilinx specifies a minimal frequency of 303 MHz for their memory interface solution [68] which makes it impossible to match the clock frequency of the memory controller to the operating frequency of the FMC110 interface. Therefore it is necessary to implement the memory controller in a different clock region than the FMC110 interface. We decided to clock the half rate controller at 200 MHz and the memory interface at its maximum frequency of 400 MHz to have the highest possible data transfer rate between the external memory and the FPGA.

Note that another clock domain not shown on Figure 11 exists which is needed by the Virtex-6 system monitor to supervise temperature and input voltage of the FPGA as described in Section 4.10. The system monitor module is synchronous to an on-chip analog-to-digital converter with a sampling frequency of 50 MHz.

4.5 Initialization Procedure of the FMC110 Mezzanine Card

In this section it is described how the different components on the FMC110 mezzanine card are initialized. This procedure has to be done whenever the card is switched on.

First of all, the registers of the complex programmable logic device (CPLD) on the mezzanine card have to be set correctly [57]. These registers can be written from the FPGA via the serial peripheral interface (SPI). The SPI communication with the CPLD is managed by the module "fmc110_cpld_ctrl" which is a submodule of the "sip_fmc110" module. The "fmc110_cpld_ctrl" allows writing to the registers of the CPLD over the Ethernet.

The CPLD controls the RF switches on the FMC110 card which are used to select the clock source for the clock tree of the FMC110 card. The clock tree is provided by the AD9517-3 chip [60] from Analog Devices, Inc. The CPLD controls the reset signal of the AD9517-3 chip. The reset signal of the AD9517-3 as well as the clock signal RF switch configuration is set via the register 0 of the CPLD. This register also controls the reset signal of the digital-to-analog converter and can be used to switch on or off the on-board 100 MHz reference oscillator.

In the experiments performed in the present thesis, the clock source is provided by the internal 1 GHz oscillator in the phase-locked loop implemented by the AD9517-3 chip. The external clock source from the dedicated clock pin of the FMC110 is forwarded to the reference clock input of the AD9517-3 which is used to control the phase of the phase-locked loop.

After the RF switches for the clock signal are set correctly and the reset signal of the AD9517-3 chip has been sent, the AD9517-3 clock tree can be initialized. Similar to the CPLD, the AD9517-3 is configured over the SPI. The module of the firmware that manages the communication with the AD9517-3 clock tree chip is called "ad9517_ctrl" which is a submodule of "sip_fmc110". As for the CPLD, the registers of the AD9517-3 are set over the Ethernet.

The AD9517-3 has a phase and frequency detector (PFD) which is used to compare

the reference clock signal to the signal of the 1 GHz oscillator in the phase-locked loop. The frequency of the clock signal from the 1 GHz oscillator is divided by a factor N at the input of the PFD whereas the signal of the reference clock is divided by another factor R. After the division, the frequencies of the input signals to the PFD have to be equal. The divisors N and R are set by registers on the AD9517-3 clock tree chip. The factor N is given by [60]

$$N = PB + A, \tag{25}$$

where P is the prescaler division factor and A and B are the values for the counters which are part of the clock divider. The setting for these values we use in the present thesis are $P = 8$, $A = 4$, $B = 12$, so that the frequency of the 1 GHz oscillator is divided by $N = 100$.

For the experiments performed in the present thesis, the reference clock is provided by a 10 MHz Rubidium frequency standard which is connected to the clock input pin of the FMC110 card. Thus the R division factor is set to $R = 1$ to match the frequency with which the signal from the 1 GHz oscillators enters the PFD through the N divider.

Alternatively, the internal 100 MHz oscillator mounted on the FMC110 mezzanine card can be used as reference clock. In that case, the R divider has to be set to $R = 10$.

The clock tree chip has 8 clock outputs where pairs of two outputs can be used to generate a differential clock signal. The first and second pair (out0, out1 and out2, out3) of outputs is connected to the analog-to-digital and digital-to-analog converter respectively. These outputs have to be configured to operate in the low-voltage positive emitter-coupled logic LVPECL mode. The third pair of outputs (out4, out5) are connected to a low-voltage differential signal (LVDS) channel which is connected to the trigger pin of the FMC110 on one side as well as to the FPGA on the other side. These outputs have to be powered down in the setting of the corresponding registers on the AD9517-3 chip when the trigger pin is used to input a signal from an external trigger source. The fourth pair of clock outputs (out6, out7) is connected via the FMC interface to the FPGA as an LVDS channel. The signal transmitted through this channel is used by the firmware of the FPGA as a reference clock. The output current of this LVDS output is set to 1.75 mA in the original firmware by 4DSP. On the contrary, tests indicate that 3.5 mA is a better setting for the current. Since the LVDS pair is terminated on the FPGA by a 100 Ω resistor, this current corresponds to a differential voltage of 350 mV which is equal to the typical value specified in the Virtex-6 DC and switching characteristics data sheet [56].

Similar to the AD9517-3 chip, the analog-to-digital converters and digital-to-analog converters are configured by writing to their registers over the SPI. Refer to the corresponding data sheets [58, 59] for a detailed description of the registers. The modules "fmc110_ads5400_ctrl" and "fmc110_dac5681z_ctrl" which are submodules of "sip_fmc110" manage the SPI communication with the analog-to-digital converter and digital-to-analog converter respectively.

Note that it would be beneficial to program a state machine on the FPGA which performs the initialization procedure described above without the need of sending the content of the registers directly over the Ethernet. This would protect the FMC110 from

possible damage caused by a possible erroneous initialization of the registers of e.g. the AD9517-3 clock tree chip which is more likely to happen when the registers are written over the Ethernet.

4.6 Parallelization and Synchronization

The rate at which the signal is sampled by the analog-to-digital converters is on the edge of todays achievable data rates for digital interfaces between FPGA. Furthermore, the processing rate of the digital signal processing on FPGA itself is limited. In Section 4.6.1 it is presented how the data can be parallelized in order to make the processing possible. Section 4.6.2 describes how the synchronization at the interface between the analog-to-digital converter and the FPGA can be achieved.

4.6.1 SerDes

In Section 4.4.3 it is explained that the maximal frequency at which the DSP resources of the Virtex-6 can be operated is less than half of the desired sampling frequency f_s of 1 GHz. To be able to process the signal at $f_\mathrm{s} = 1\,\mathrm{GHz}$, the digital signal processing on the FPGA thus has to be done in parallel. The Virtex-6 architecture offers a specialized primitive called "SerDes" (Serializer/Deserializer) which converts the serial data stream at the input of the FPGA into a parallel bus. For each of the 12-bit samples which are streamed from the analog-to-digital converter to the FPGA at a rate of 1 GHz, a SerDes is instantiated which maps 8 successive samples into a word of 8 bits synchronous to a 125 MHz clock. This situation is depicted in Figure 12.

As part of the parallelization of the data stream by the SerDes, the incoming samples have to be stored in flip-flops. Thus each SerDes has two clock inputs: one clock input requires a source-synchronous 500 MHz clock with which the data is registered at the input of the SerDes at both the rising and falling edge. Thus the SerDes is actually a double data rate (DDR) interface. To clock the parallel output, another input clock with a frequency of 125 MHz is required. The two clock signals must have a fixed phase relation which remains constant over time so that the event of a rising edge at the beginning of one period of the 125 MHz clock signal takes place at the same time as the rising edge of the 1 GHz clock signal. For this to be achieved, either both signals have to stem from the same phase-locked loop on the Virtex-6 FPGA or the slow clock signal has to be derived from the fast source-synchronous clock signal by a clock-dividing regional clock buffer (BUFR) [69]. Only the former method makes it possible to clock all of the digital signal processing operations with the same source-synchronous clock signal. Being able to do this is important, especially if feedback signals have to be sent to the digital-to-analog converter based on the data received from the analog-to-digital converters. Our implementation of the firmware therefore makes use of a phase-locked loop by instantiation of an MMCM that receives a reference clock signal from the FMC110 as described in Section 4.4.3. The MMCM is instantiated in the VHDL module "fmc110_ref_clock" which is a submodule of "fmc110_if".

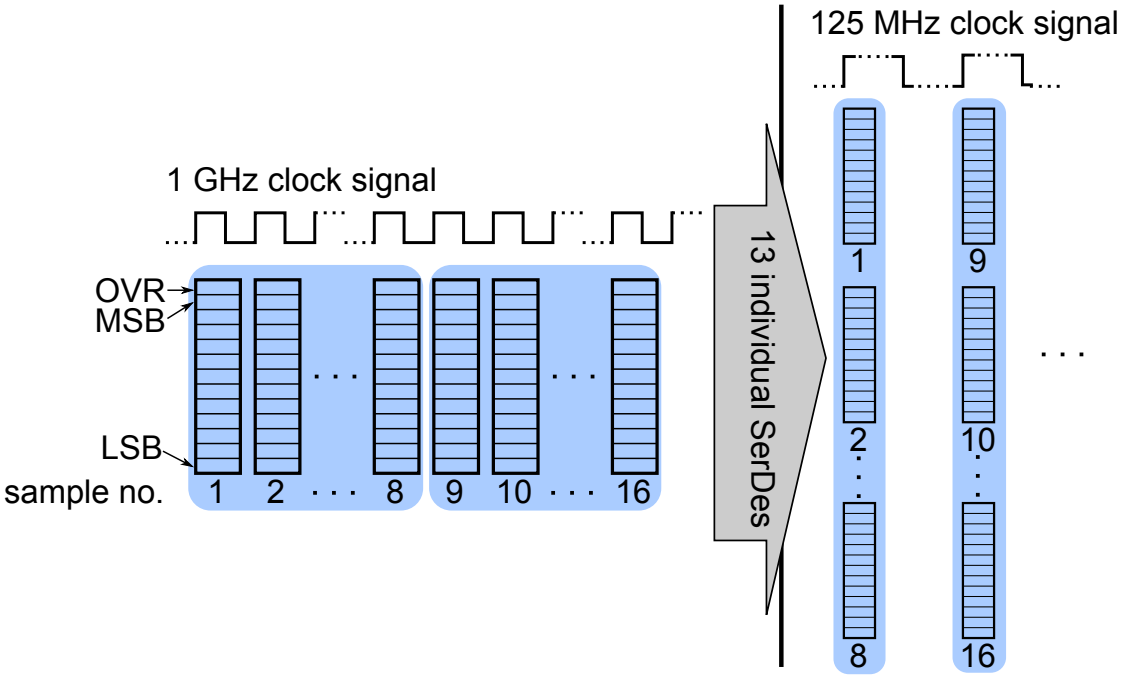


Figure 12: Parallelization of the 13 bit signal received from the analog-to-digital converter using the SerDes primitives of the Virtex-6 FPGA. The digital samples are represented by 12 bit words and an "over range" (OVR) bit which indicates whether the absolute voltage of the analog signal was above or below the full-scale voltage of the converter. Before the bits of the samples arrive at the input of 13 individual SerDes, the samples are serial with respect to the 1 GHz clock, i.e. in each clock cycle only one sample arrives at the FPGA. Each SerDes outputs 8 bits corresponding to 8 successive samples in parallel synchronous to a 125 MHz clock signal.

4.6.2 Bit Align Machine

A problem that arises when data is to be received from the analog-to-digital converter is that the phase relationship between the sender clock with which the analog-to-digital converter is clocked and the receiver clock with which the SerDes is clocked is not known at compile time. In addition to that, the arrival time of the data at the SerDes can vary due to temperature and voltage fluctuations [70]. Thus, unlike the usual situation where data is sent across synchronous elements within the FPGA which is described in Section 4.4.1, the compile tools cannot predict the time at which the data is stable at the input of the SerDes with respect to the time at which the clock event arrives at the SerDes. Therefore the compile tools cannot determine whether the setup and hold times of the flip-flops inside the SerDes are met.

The goal of the bit align machine is to find the optimal clock-to-data relationship so that both setup and hold times are fulfilled as depicted in Figure 13. The time interval within which the setup and hold time requirements are fulfilled is called "data eye". The bit align machine delays the data with respect to the clock event in such a way that the clock events take place at the center of the data eyes. The delay of the data can be adjusted by using the IODELAYE1 primitive of the Xilinx Virtex-6 FPGA which gives access to 32 delay taps.

For the tap delays, a reference clock is needed. The 200 MHz reference clock is derived from the same MMCM as introduced in Section 4.6.1. This reference clock has to be supplied to the so called IDELAYCTRL primitive which controls all IODELAYE1 primitives within the same group specified by the IODELAY_GROUP attribute. For a 200 MHz clock the average delay between each tap amounts to 78 ps.

Like the SerDes, the bit align machine is instantiated for every of the 12 bits that are sent in parallel from the analog-to-digital converter to the FPGA as well as for the additional over-range bit. The same structure is used for both analog-to-digital converters.

In order to detect whether the clock events take place inside the data eyes, a test is needed to check whether the data has been transferred successfully or not. For this purpose, the ADS5400 analog-to-digital converters can be configured to send a pseudo random bit sequence (PRBS) [58] to the FPGA. This is done by setting bits 6 and 7 (counted from 0) of register 0x06 of the ADS5400 to 0 and 1 respectively. For the generation of the bit sequence, a linear feedback shift register (LFSR) with 7 bits is used. The shift register maintains a word of 7 bits which is shifted in every clock cycle e.g. to the left. Thus in every clock cycle, one bit is shifted out of the register. This is the bit that is sent to the FPGA. Supposing the bits are shifted to the left, the two leftmost bits are taken as the input of an XOR gate. The result of the XOR operation on these two bits is fed back as input from the right into the LFSR in the subsequent clock cycle. In this manner, a PRBS pattern is generated with a period of $2^7 - 1 = 127$ clock cycles.

On the FPGA, this pattern is then tested in a PRBS checker module programmed in VHDL based on a Xilinx Application Note by D. Riccardi and P. Novellini [71]. The PRBS checker continuously compares the 8-bit output of every instantiated SerDes to

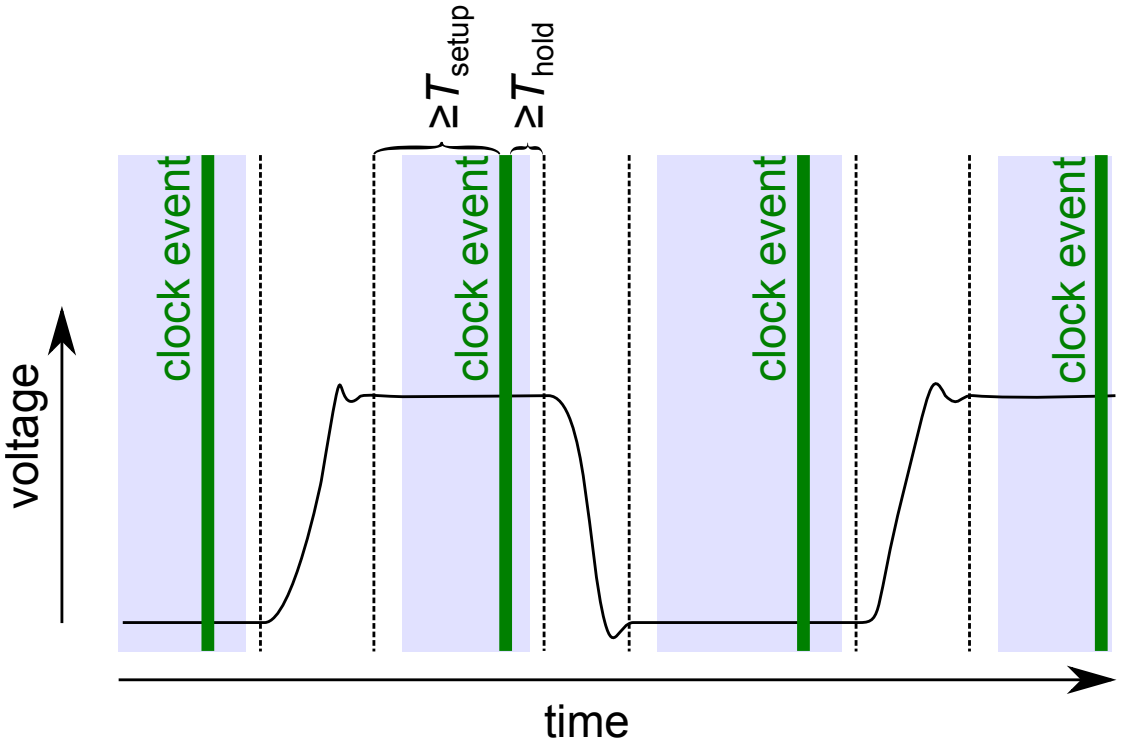


Figure 13: Illustration of the clock-to-data relationship. The signal voltage is schematically drawn as a function of time. The time intervals during which the voltage remains constant are circumscribed by dashed lines. The green bold vertical lines indicate the clock events. The time during which the voltage is constant before and after the clock event has to be larger or equal to the setup time T_{setup} and hold time T_{hold} respectively. The time frame within which these requirements are fulfilled, the so called "data eye", is indicated by the blue shaded boxes.

what is expected from the previous 8 bits that were seen. When the patterns do not match, the PRBS checker counts the number of bits in which the two patterns did not match. These error counts are then accumulated over a certain amount of clock cycles to get the total number of errors which occurred during n clock cycles for every instantiated SerDes.

The total number of errors encountered during n clock cycles is used as a criterion for the determination of whether the clock event takes place in the data eye or not. Investigations on the optimal number of clock cycles over which the error count should be integrated, indicate that the averaging should be done for a very long time, i.e. on the order of a second. In that way, errors are taken into account that are due to fluctuations that happen on this time scale. In the present version of the PRBS checker, the summation is done for $n = 2^{27}$ clock cycles which corresponds to a duration of $n(8\,\mathrm{ns}) \approx 1\,\mathrm{s}$.

For the following considerations it is important that the bit align machine operates periodically, i.e. that a decrease of the tap delay when it is initially 0 means that the tap delay is increased by an amount of tap delays which corresponds to the clock period of the 500 MHz clock.

The bit align machine is started by writing a '1' to the third bit of the command register in the "ads5400_phy_sp" module which is a submodule of "fmc110_if". When the bit align machine is started, the PRBS checker is invoked to check whether the initial tap delay, which is set to 0 in our firmware, already corresponds to the clock edge being aligned to a data eye. The bit align machine only considers the clock edge to be inside the data eye when the PRBS checker reports no errors during the entire checking duration. If the clock event happens inside the data eye, the bit align machine shifts the data eye by decreasing the tap delay. The PRBS check is repeated after each change of the tap delay until the PRBS checker reports a non-zero number of errors during the check duration. When errors occur after decreasing the tap delay, the tap delay value obtained after the decrease is marked as the first edge of the data eye.

In the case when the initial tap delay corresponded to the situation where the clock edge is outside the data eye, i.e. PRBS errors occur with the initial alignment, the tap delay is increased and the check is repeated until the PRBS error sum becomes 0. Unlike to the former case, the *previous* tap delay value is then marked as the first edge of the data eye.

In the next stage of the bit alignment, the tap delay is increased until the number of errors detected by the PRBS checker becomes non-zero again which indicates that the second edge of the data eye has been reached. The number of increases of the tap delay in this second stage is called window size. The bit align machine can now shift back the data eye by decreasing the tap delay by an amount of half the windows size such that the clock event takes place in the center of the data eye.

When a data eye has been found for all of the 13 channels (12-bit word and over-range bit) coming from the analog-to-digital converter, it usually occurs that the delay in terms of entire clock cycles of the sampling clock between the 13 channels does not match for all signals. In that case, although the bit patterns are correct for every single input channel, some bit patterns are shifted with respect to each other. This problem is solved

by declaring one bit align machine as the "master". For every "slave" it is attempted to align its bit pattern to the one of the master by shifting the 8 bits that exit the SerDes and by introducing delays with filp-flops that are synchronized to the 125 MHz clock so that it is possible for samples to "slip" from one 8-bit word to the other.

This so called bit slip operation is repeated until either the bit pattern always matches with the one from the master or a certain threshold for the delay introduced by the bit align machine is reached. When a slave reaches the delay threshold, it instructs the master to perform a bit slip operation by itself. After the bit slip operation of the master, the bit slip of the slaves is reset and adjusted again to the reference pattern sent by the master.

Since the shift between the bit patterns of the individual channels is usually small, this algorithm always converges after a certain amount of bit slip operations to a situation where in every clock cycle, the 8-bit patterns at the output of the bit align machine match for every of the 13 channels.

Note that the PRBS pattern generated by the analog-to-digital converter has the property that a sequence of 8 bits only repeats after 127 clock cycles of the 125 MHz clock. For safety it is checked multiple times whether the 8-bit pattern from the slave matches the one from the master. If the patterns always match, it is concluded that the slave is aligned to the master. If the patterns never match, another bit slip operation is performed. If the patterns sometimes match, it is concluded that the clock-to-data alignment was not successful and the entire bit alignment procedure is restarted.

The state of the bit align machine can be read out via the command register of the "ads5400_phy_sp" module. Furthermore, the total number of errors obtained during the PRBS check can be read out from a separate register of the "ads5400_phy_sp" module.

Note that even after the bit alignment has been performed we observe mismatches between the test patters of the channels which happen roughly once every 4 seconds. The drawback of the above described method is that the PRBS pattern has to be turned off when real data has to be measured with the analog-to-digital converter.

A Xilinx application note by M. Defossez [72] describes a possible solution to this problem. The application note describes an alignment of the clock-to-data relationship during run-time based on the clock signal sent from the analog-to-digital converter to the FPGA. Although the method of Defossez has not been implemented in the firmware at the time of writing the present thesis, it is a promising add-on to the above described bit alignment procedure which could possibly avoid alignment errors on long timescales.

4.7 Digital Signal Processing

As motivated in Section 4.1, the configurable digital signal processing is the main advantage of FPGA-based measurement solutions. From the overview of the firmware provided in Section 4.3 it can be seen that the digital signal processing is done in the module "sip_user_app_component". This module is special in the sense that its main component, the "user_app_component" itself, is not programmed in VHDL but in the graphical programming environment Simulink which is well suited to design digital signal processing systems. This is done with aid of the Xilinx System Generator for DSP which

offers the basic building blocks such as logic operations, adders, multipliers, etc. as well as blocks for more complicated digital signal processing operations like the fast Fourier transform. The "user_app_component" is integrated into the rest of the firmware in such a way that the designer of the digital signal processing system is provided with a versatile interface to the different components of the firmware. Figure 14 shows the top-level module of the graphically programmed part in the user_app_component.

The top-level module shown in Figure 14 reflects the basic data flow which is generic in the sense that it is common to a variety of digital signal processing applications. Shown on the left end of Figure 14 are the first blocks in the data flow. These blocks are signal generators which simulate input signals coming from other blocks of the firmware. The simulation blocks allow for simulation of the digital signal processing directly in Simulink independently of the rest of the firmware.

The simulated signals are then forwarded to the input blocks shown in the second column of Figure 14. In the input blocks, the simulated signal is either just forwarded or else replaced with the real input signals that come from outside the graphically programmed part of the user_app_component depending on whether the design is only simulated or compiled for the actual configuration of the FPGA.

The actual signal processing takes place in the module denoted as "Configurable Digital Signal Processing" which is a generic module which allows the selection of different implementations. In this way it is possible to program different digital signal processing applications while using the same input and output blocks.

After digital signal processing, the processed signals are sent to the output blocks which return the signals to the higher level module outside the graphically programmed part of the firmware. Additionally, the simulated output is recorded at the scope shown on the right side of Figure 14.

4.7.1 Correlator Application

The correlator application is a particular implementation of the generic configurable digital signal processing block presented in the introduction to Section 4.7. Its purpose is to compute the power spectrum of the measured signal in order to e.g. observe Mollow triplets as shown in Section 2.7. The definition of the power spectrum of a time-dependent complex signal $S(t)$ is:

$$P(\nu) \equiv \mathcal{F}[S(t)](\nu)\,(\mathcal{F}[S(t)](\nu)^*)\,, \tag{26}$$

where $\mathcal{F}$ denotes the Fourier transform and ν denotes the frequency.

This expression is evaluated on the FPGA in four successive stages. The first stage is the digital down conversion of the signal as described in Section 4.7.3. Note that as explained in Section 4.7.3, the digital down conversion only shifts the power spectrum along the frequency axis. It is therefore in principle not necessary to perform a digital down conversion for the calculation of the power spectrum. Nevertheless, the digital down conversion with one quarter of the sampling frequency costs only one additional logic operation per signal channel. We therefore implemented the down conversion for

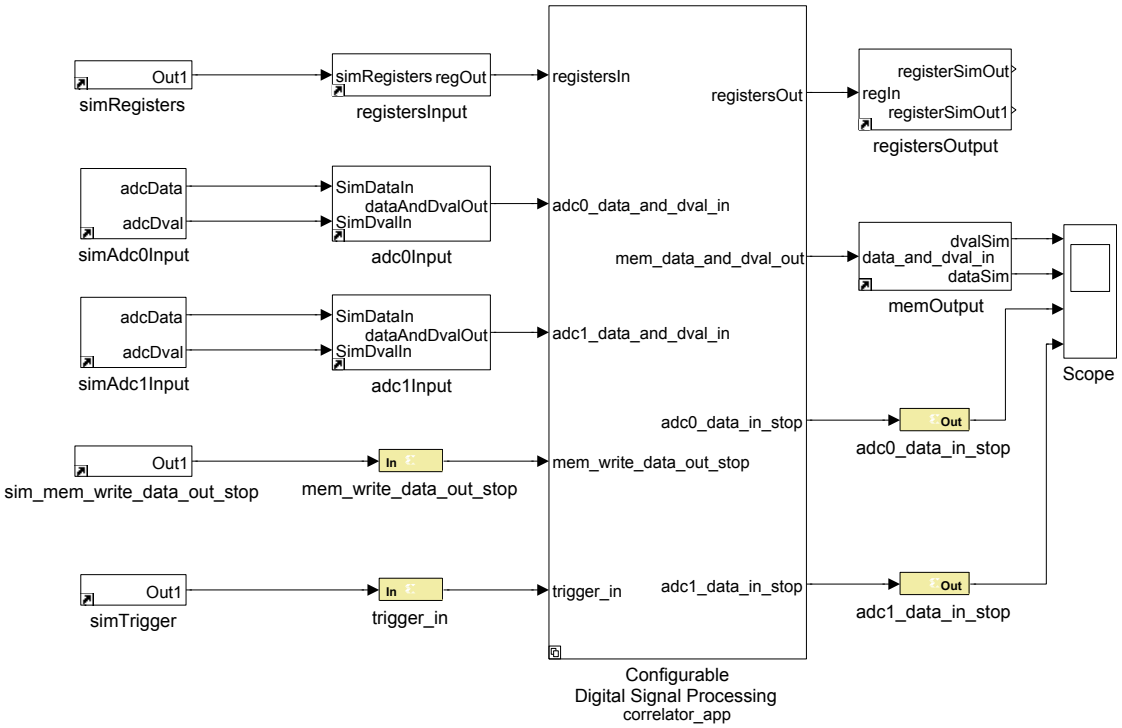


Figure 14: The top-level module of the graphically programmed core of the module "sip_user_app_component". The design is divided into different subsystems represented by the white boxes. The arrows represent buses of several bundled digital signal channels where the direction of the arrows signifies the direction of the signal flow. The yellow boxes represent input and output ports which are used to interface with the rest of the firmware. Note that the subsystems with the suffix "Input" and "Output" also contain input and output ports respectively. The subsystem labeled "Configurable Digital Signal Processing" is a template that allows one to choose different implementations. In this picture, "correlator_app" is the actual implementation that has been chosen.

convenience and as a template for future applications where e.g. the signal will be low-pass filtered after the digital down conversion.

After the down conversion, the Fourier transform is performed on the samples which arrive in parallel due to the deserialization described in Section 4.6.1.

The third stage is the complex multiplication of the Fourier transformed signal with its complex conjugate to get the absolute square value of the Fourier coefficients. This is done using the dedicated multiplier resources of the Xilinx Virtex-6 FPGA.

The absolute square values of the Fourier coefficients are then averaged in the last stage which is described in Section 4.7.4.

The reader may wonder why this application is called "Correlator". As argued in Section 2.7, in order to get the first-order autocorrelation function in the time domain, the only thing that has to be done is inversely Fourier transform the power spectrum. Due to the linearity of the inverse Fourier transform, this can be done after the averaging. Therefore this operation is not time-critical as it does not have to be performed for every single power spectrum. After the averaged data has been sent to the host computer, the inverse Fourier transform can be performed using the CPU of the host computer.

4.7.2 Parallel Fast Fourier Transform

Computing the fast Fourier transform (FFT) on the FPGA together with averaging is key to the efficient computation of correlation functions in long enough time windows. As shown in this section, the FFT can be done in a pipelined manner on the FPGA, which makes it possible to obtain good statistics by averaging over a large number of transformed sequences with a minimum waiting time between the acquisition of the individual traces. A major challenge with this procedure is that in every cycle of the processing clock, 8 samples arrive in parallel from the SerDes described in Section 4.6.1.

The Xilinx System Generator for DSP offers a module which computes the FFT of a sequence of complex or real valued samples in a pipelined manner. However, this building block assumes that in every clock cycle only one sample approaches the input at a time. When it is applied on only one of the 8 samples that have to be processed in every clock cycle of the 125 MHz clock, this corresponds to a so called down sampling after which the bandwidth of the computed spectrum is only 62.5 MHz instead of 500 MHz and the information about the other samples is lost.

Here we show how it is possible to compute the discrete Fourier transform of the full time sequence of samples with a bandwidth 500 MHz by computing 8 parallel FFTs. The derivation follows the same principle used by Cooley and Tuckey in their version of the fast Fourier transform algorithm [73]. The idea is to repeatedly divide the samples x_k ordered with ascending index k into groups of samples with even and odd indexes respectively. Repeating this procedure 3 times, 8 sets of indexes are obtained as shown in Figure 15.

We use the following definition of the discrete Fourier transform

$$X_k = \sum_{n=0}^{N-1} x_n \, \mathrm{e}^{-2\pi i k n/N}, \qquad k = 0, 1, 2, 3, \ldots, N, \tag{27}$$

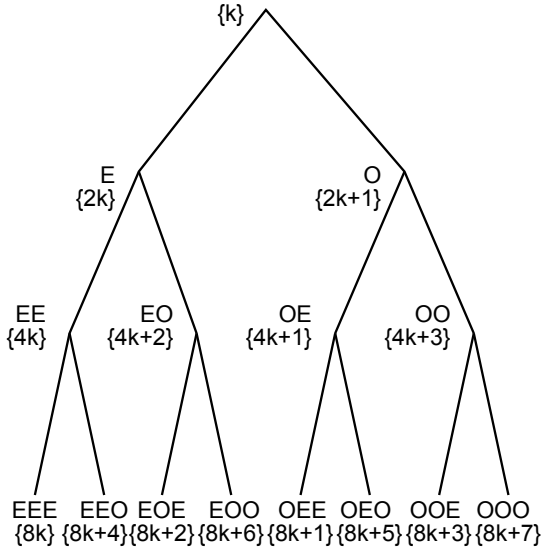


Figure 15: Recursive division of the indexes of the samples into sets of even and odd indexes. The labels E and O stand for even and odd respectively. The set notation is short-handed where $\{k\}$ is the abbreviation of the set defined as $\{k \mid k \in \mathbb{N}_0 < N\}$ where N is the total number of samples and $\mathbb{N}_0$ is the set of natural number including 0.

where N is the length of the discrete Fourier transform. We now split this sum into two sums, one summation over all even indexes and the other summation over all odd indexes

$$\begin{aligned} X_k &= \sum_{n=0}^{N/2-1} x_{2n}\, \mathrm{e}^{-4\pi i k n/N} + \sum_{n=0}^{N/2-1} x_{2n+1}\, \mathrm{e}^{-2\pi i k(2n+1)/N} \\ &= \sum_{n=0}^{\tilde{N}-1} x_{2n}\, \mathrm{e}^{-2\pi i k n/\tilde{N}} + \mathrm{e}^{-2\pi i k/N} \sum_{n=0}^{\tilde{N}-1} x_{2n+1}\, \mathrm{e}^{-2\pi i k n/\tilde{N}}\,. \end{aligned} \tag{28}$$

We note that the two sums are again two discrete Fourier transforms of length $\tilde{N} = N/2$. To compute the discrete Fourier transform for $k \geq N/2$, Equation (28) is evaluated for $\tilde{k} = k - N/2$ as follows

$$\begin{aligned} X_{\tilde{k}+N/2} &= \sum_{n=0}^{\tilde{N}-1} x_{2n}\, \mathrm{e}^{-2\pi i(\tilde{k}+\tilde{N})n/\tilde{N}} + \mathrm{e}^{-2\pi i(\tilde{k}+\tilde{N})/N} \sum_{n=0}^{\tilde{N}-1} x_{2n+1}\, \mathrm{e}^{-2\pi i(\tilde{k}+\tilde{N})n/\tilde{N}} \\ &= \sum_{n=0}^{\tilde{N}-1} x_{2n}\, \mathrm{e}^{-2\pi i \tilde{k} n/\tilde{N}} - e^{-2\pi i \tilde{k}/N} \sum_{n=0}^{\tilde{N}-1} x_{2n+1}\, \mathrm{e}^{-2\pi i \tilde{k} n/\tilde{N}}\,. \end{aligned} \tag{29}$$

From this we see that the computation of X_k for $k \geq N/2$ differs from the computation of X_k for $k < N/2$ only in the prefactor of the discrete Fourier transform over the samples

with odd indexes. Thus we only have to compute the Fourier transform of length $N/2$ for $k < N/2$ one time for all even and the other time for all odd samples.

To summarize this result we denote the discrete Fourier transform over all even samples as E_k and the Fourier transform over all odd samples as O_k. We have

$$X_k = \begin{cases} E_k + \mathrm{e}^{-2\pi i k/N}\, O_k & , k < N/2 \\ E_{k-N/2} - \mathrm{e}^{-2\pi i (k-N/2)/N}\, O_{k-N/2} & , k \geq N/2 \end{cases}, \qquad k = 0, 1, 2, 3, \ldots, N. \tag{30}$$

By recursively applying Equation (30), we can express the discrete Fourier transform of a sequence of N samples in terms of 8 discrete Fourier transforms over the sequences of samples with indexes from one of the following 8 different sets

$$I_m \equiv \{8k + m \mid k = 0, 1, 2, 3, \ldots, N/8\}, \qquad m = 0, 1, 2, 3, 4, 5, 6, 7. \tag{31}$$

The sequences of samples with indexes of one of these sets correspond to the sequence obtained by taking in each clock cycle a particular sample of the 8 samples that arrive in parallel.

4.7.3 Digital Down Conversion

In the same manner as analog down conversion shifts the spectrum of an analog signal, digital down conversion can be employed to shift the discrete spectrum of a digital signal by a frequency offset f_c. How this is done is explained in the present section after a motivation of the use of digital down conversion.

In Section 3 it is explained that analog down conversion can be used to shift the main frequency component of the analog signal to an intermediate frequency (IF) before the signal is digitized. Digital down conversion can be used to shift this IF signal component further down to 0 MHz. In this way it is possible to obtain for example the power of the original IF signal component by determining the power of the DC component of the digitally down-converted signal without having to do a Fourier transform. It follows from the definition of the discrete Fourier transform given by Equation (27), that the Fourier coefficient which corresponds to the frequency bin centered around the DC component is given by

$$X_0 = \sum_{n=0}^{N-1} x_n\, \mathrm{e}^0 = \sum_{n=0}^{N-1} x_n. \tag{32}$$

Thus the Fourier coefficient of the DC component is simply obtained by summing over all data points that belong to the digital signal. The absolute square value $|X_0|^2$ is proportional to the power of the DC component, whereas the absolute value $|X_0|$ is proportional to the amplitude. The argument $\arg[X_0] = \tan^{-1}(\mathrm{Im}[X_0]/\mathrm{Re}[X_0])$ corresponds to the phase of the DC signal. Digital down conversion followed by summing over the (complex) data points can therefore be used to compute amplitude and phase of the signal at frequency f_c.

In the following it is explained how digital down conversion works. Let the discretized signal be given by the data points x_n in time-domain, where n is the index of the sample.

Complex digital down conversion consists of the multiplication of the time-dependent discretized input signal with a complex-valued factor as follows [23]

$$\tilde{x}_n = x_n \, \mathrm{e}^{-2\pi i n f_\mathrm{c}/f_\mathrm{s}}, \tag{33}$$

where $\tilde{x}_n$ is the down converted signal.

Let r be the ratio $r = f_\mathrm{s}/f_\mathrm{c}$ between sampling frequency and down conversion frequency. The length of the discrete Fourier transformation N is chosen in such a way that N/r takes on an integer value. The effect of the digital down conversion of the signal x_n can be seen in the down converted signal $\tilde{x}_n$ given by Equation (33). For this $\tilde{x}_n$ is plugged into the definition of the discrete Fourier transform from Equation (27)

$$\begin{aligned}
\tilde{X}_k &= \sum_{n=0}^{N-1} \tilde{x}_n \, \mathrm{e}^{-2\pi i k n/N} \\
&= \sum_{n=0}^{N-1} x_n \, \mathrm{e}^{-2\pi i n/r} \, \mathrm{e}^{-2\pi i k n/N} \\
&= \sum_{n=0}^{N-1} x_n \, \mathrm{e}^{-2\pi i (k+N/r) n/N} \\
&= X_{(k+N/r \mod N)}.
\end{aligned} \tag{34}$$

The modulo operation in the index of $X_{(k+N/r \bmod N)}$ follows from the 2π periodicity of the complex exponential in the definition of the Fourier transform. From Equation (34) it can be seen that the complex digital down conversion circularly shifts the discretized spectrum by an amount of $N/r = N f_\mathrm{c}/f_\mathrm{s}$ towards lower indexes.

The digital down conversion for $f_\mathrm{c} = f_\mathrm{s}/4$ can be implemented particularly efficient on an FPGA since the factor $\mathrm{e}^{-2\pi i n f_\mathrm{c}/f_\mathrm{s}}$ from Equation (33) takes on the values

$$\mathrm{e}^{-2\pi i n f_\mathrm{c}/f_\mathrm{s}} = \mathrm{e}^{-2\pi i n/4} = \cos\left(\frac{2\pi n}{4}\right) + i \sin\left(\frac{2\pi n}{4}\right) = \begin{cases} 1 & , n = 4k \\ -i & , n = 4k+1 \\ -1 & , n = 4k+2 \\ i & , n = 4k+3 \end{cases}, \tag{35}$$

where k is an integer. Complex multiplication of the measured real-valued samples by this factor is particularly simple to implement as only the sign is changed and the real and imaginary part are exchanged in some cases. On the FPGA, 8 successive samples are processed in parallel in every clock cycle and in a pipelined manner as described in Section 4.6.1. Thus the complex digital down conversion of the real signal coming from an analog-to-digital converter can be implemented as depicted in Figure 16.

4.7.4 Averaging

The absolute square value of the Fourier coefficients obtained by the algorithm described in Section 4.7.2 has to be averaged due to the noise in the measurement data. The entity in our firmware that performs this task is called the “averager”.

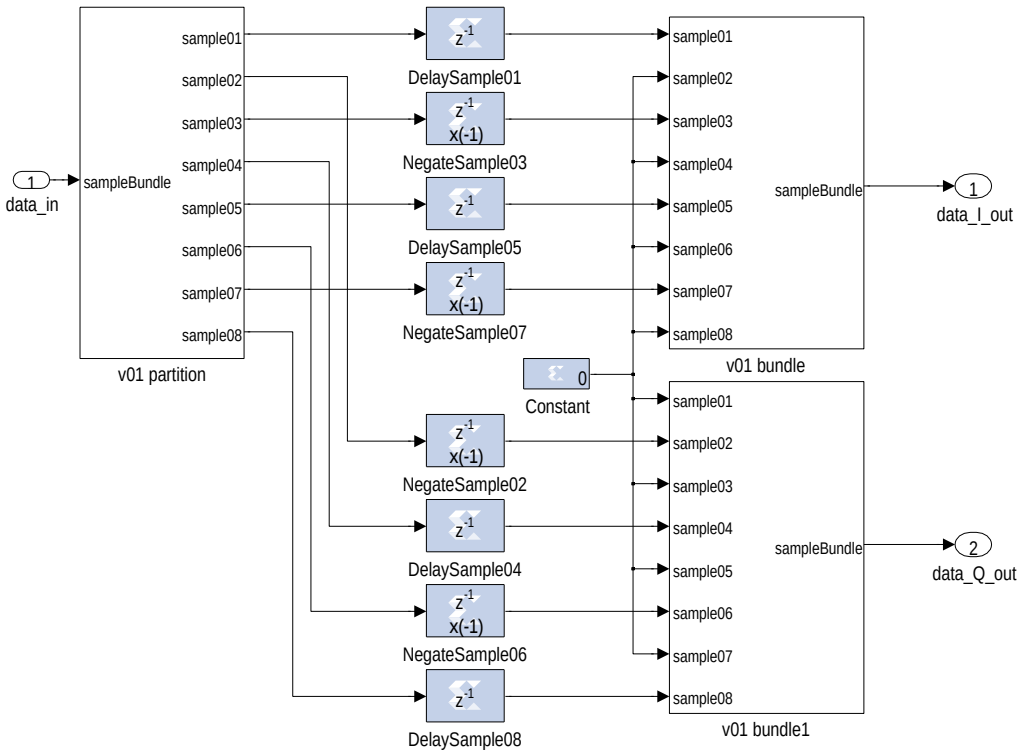


Figure 16: Digital down conversion of a real signal with a down conversion frequency of $f_c = f_s/4$ as it is implemented in the digital signal processing of the 8 parallel digital signals. The block called "v01 partition" unbundles the data_in bus into 8 smaller buses each representing a particular parallel sample. The sign of some samples is flipped by the "Negate" operations. Every second sample of either the real, ("data_I_out") or imaginary ("data_Q_out") output are set to the constant 0. The "bundle" operation concatenates the 8 parallel samples to a single bus. In order to prevent the violation of timing constraints, the signal is delayed by one clock cycle in the boxes labeled with z^{-1}.

The averaging of these values is performed by using the dual port RAM of the Xilinx System Generator block set. The dual port RAM uses block RAM on the FPGA which makes it possible to read out values at the same time as writing to the RAM. Each address of the dual port RAM corresponds to a certain frequency bin of the power spectrum. When the fast Fourier transform is activated, the averager receives the power values and indexes for 8 specific frequency bins.

Thus the data coming from the FFT is delayed for a number of clock cycles needed to read the stored sum of the corresponding 8 frequency bins out of the RAM so that these values arrive at 8 independent adders at the same time as the newly obtained values. This happens in the usual pipelined manner so that effectively in every clock cycle 8 values are added with the previously obtained sum and stored in the RAM.

In order to store all of the 8 parallel samples and to prevent an overflow of the fixed point numbers, the data width W_{RAM} of the values stored in the RAM has to be obey the following rule

$$W_{\mathrm{RAM}} = 8(W_{\mathrm{inc}} + \log_2 D_{\mathrm{avg}}), \tag{36}$$

where W_{data} is the data width of the increments and $D_{\mathrm{avg,max}}$ is the maximum average depth, i.e. the maximal number of measurement shots over which the average should be taken. In order to be able to choose the average length without the need to recompile the firmware, $D_{\mathrm{avg,max}}$ should be set to a large value.

The actual average depth can be set by the host computer by writing to a register in the address space of the user app component through the Ethernet interface. Via this register it is also possible to activate a special mode of the averager, called the "diff mode". In the diff mode, the sign with which the newly obtained values are added to the value stored in the RAM is flipped in every measurement shot. This mode is intended to perform so called background subtraction by switching the input signal on and off in every clock cycle so that the noise power spectrum obtained when the signal is switched off is subtracted from the part of the spectrum that represents the signal of interest.

4.8 Memory Interface

When very long time traces have to be measured, it is necessary to swap the measured data from the FPGA to its external RAM. With the interface provided by the module "sip_mem_if", the "user_app_component" can write data to the external RAM which can then be read out over the Ethernet by the host computer.

The central unit of the memory interface is the state machine implemented in the submodule "mem_if_memc". The state machine allows to read and write sequences of addresses making use of the burst feature of the Xilinx DDR3 memory controller which minimizes the waiting times between read/write accesses at different addresses as explained in the following. The DDR3 RAM has a word width of 64 bit, meaning that there is one 64 bit word per address. Because the double data rate feature allows to read two words per clock cycle and since the direct communication with the memory takes place at 400 MHz, the memory interface can read or write 4 words at consecutive addresses in each clock cycle of the 200 MHz clock. The waiting time is minimized when a total of 8 words are read/written at consecutive addresses. The sequence of 8 words

is called a "burst". If instead only one word at one address is to be written, then a mask can be supplied to the Xilinx memory controller which tells the controller which words of the burst are to be written to the memory and which are not. The mask can, however, not yet be used with the implementation of the memory controller at the time of writing this thesis. Also notice that only the burst mode allows to write multiple addresses in a sequence. If the burst mode is not used, there will be a waiting time (also called turnaround) between successive read/write commands to the DDR3 RAM which forbids unbuffered continuous read/write sequences. This stands in contrast to the so called zero-bus turnaround (ZBT) RAM that was available on the development board used for the previous Virtex-4 FPGA which allowed to write any sequence of read/write commands at different addresses without waiting time between the successive commands and with a well defined latency.

The memory interface contains a first-in-first-out (FIFO) memory which manages the crossing from the 125 MHz clock domain of the user app component to the 200 MHz memory interface clock as described in Section 4.4. This generated FIFO module is called "mem_if_fast_write_fifo".

To manage the crossing from the memory interface clock to the 125 MHz clock domain used by the Ethernet, a handshaking method is used. The handshaking is done between the main state machine in the module "mem_if_memc" on the side of the 200 MHz clock domain and another state machine implemented in the module "mem_if_read_data_sync" on the side of the 125 MHz clock domain.

Similar to the "user_app_component", the module "sip_mem_if" is controlled by the host computer by writing commands to corresponding registers over the Ethernet interface. The command register has the address 0 in the address space of the memory interface. In its present form there are three commands: "no operation (NOP)", "read sequence" and "write sequence". The latter two commands cause the memory interface to read or write the values stored at a sequence of subsequent addresses. The starting address and the length of the sequence is written by the host computer to the registers with address 1 and 2 respectively. The commands are only executed if the memory interface is in the wait state and if the value of the command register is changed. The "no operation" command causes the memory to remain in the wait state.

The read-only registers with addresses 3, 4 and 6 give information about the state of the different parts of the memory interface. Register 3 gives information on how many addresses have been read as a consequence of the read sequence command that has been issued most recently. Register 4 contains the state of the synchronization state machine in the module "mem_if_read_data_sync". Register 6 contains the value of a counter that counts how many times the FIFO implemented in "mem_if_fast_write_fifo" encountered an overflow, i.e. a write command when there is no storage space left in the FIFO.

Furthermore the register with address 5 allows the selection of a "write pattern". In the default setting, where the write pattern selection register is set to 0, the data which is sent to the memory interface from the user app component is written to the memory when a "write sequence" is issued. When the write pattern selection register is set to a non-zero value, a generated pattern is written to the memory whenever a "write sequence" command is issued. In that way, the write and read process can be

tested and the content on the external RAM can be deleted. The patterns that are available for this purpose, are the "zero pattern" and the "address pattern". The zero pattern corresponds to overwriting the content of the memory at each address of the write sequence with a value of 0. In contrast, the address pattern corresponds to writing at each address of the write sequence a value that equals to the corresponding address.

4.9 Application Programming Interface (API)

In order to read and write registers of the FPGA and in order transfer the measured data to the host computer, a counterpart to the firmware is needed on the host computer. The research group in which the present thesis is written, uses a custom made measurement software called CleanSweep which is based on the LabVIEW software by National Instruments. In order to integrate the control software for the FPGA-based measurement instrument into the LabVIEW environment, a dynamic linked library (DLL) has been programmed in `C++`. The DLL offers an application programming interface (API) which consists of a set of functions that can be used as building blocks in LabVIEW or also in a stand-alone application. The API has been integrated into CleanSweep to be able to setup and perform measurements with the new FPGA. Furthermore a stand-alone `C++` console application has been programmed which serves as a test of the API and can also be useful to implement and test new features of the firmware. The API functions are declared in the `C++` header file called "FMC110Lib.h". In the present section, the most important functions offered by the API are described.

The first step when using the API is to create an "address handle" which is a data structure that contains the addresses of the individual submodules of the firmware. The handle is created using the function "FMC110_createHandle". This function allocates the needed memory on the host computer to store the data structure and returns a pointer to the corresponding memory location. Within the LabVIEW program, this pointer has to be treated as an unsigned integer since the data type of the handle is not known inside the LabVIEW program. The data width of this integer is either 64 bit or 32 bit depending on the processor architecture of the host computer. The function "FMC110_createHandle" takes an identifier of the physical Ethernet interface to which the FPGA is connected as an argument. A list of the available Ethernet interfaces and the corresponding identifiers can be displayed using the function "FMC110_printDevicesEnumeration". Since this function writes the list of Ethernet interfaces to the console, it can only be used as part of the console application at the time of writing this thesis. Every function of the API except the two functions described above takes the pointer to the address handle as an argument. All functions of the API except "FMC110_createHandle" return an error code which is an integer that indicates whether the command was successful or not. A value of 0 means that the command was successful whereas negative values are used to indicate different sources of error.

The second step is to establish an Ethernet connection to the FPGA and fill the handle with the addresses of the submodules of the firmware. This is done using the function "FMC110_openCard". Note that since the Ethernet connection to the FPGA is established through a proprietary network protocol by 4DSP which directly builds

on the Media Access Control (MAC) protocol, the FPGA has to be directly connected to a network interface of the host computer using a crossover Ethernet cable, meaning that the pair of lines used to transmit data cross over the pair of lines that are used to receive data. Note that in principle it is also possible to connect the FPGA to the same Ethernet switch or hub as the host computer is connected to. In this case, however, we observed data loss which is possibly due to timing errors in the network protocol by 4DSP when the FPGA is not directly connected to the host computer.

The network connection to the FPGA is closed by the function "FMC110_closeCard" which also releases the memory which is allocated for the address handle.

The initialization of the different components of the FMC110 card as described in Section 4.5 is invoked using the function "FMC110_init". Refer to the comments on the function declaration in the file "FMC110Lib.h" for an explanation of the arguments this function takes.

The source of the trigger signal sent to the user app component can be selected using the function "FMC110_selectTriggerSource". This function writes the trigger selection to the control register at address 1 of the "fmc110_ctrl" module which is a submodule of "sip_fmc110". The default trigger source is a software trigger which can be sent from the host computer to the FPGA using the API function "FMC110_sendSoftwareTrigger". Using the function "FMC110_selectTriggerSource", it can be selected to trigger on the rising edge, falling edge or both edges of the digital signal from an external trigger connected to the trigger input port of the FMC110 card. Furthermore it is possible to trigger on the occurrence of a specific 8-bit pattern when one of the analog-to-digital converters is in the PRBS mode described in Section 4.6.2.

In order to read or write the registers of the "user_app_component", there are application specific functions in the API. The function that is used to initialize the correlator application described in Section 4.7.1, is called "user_app_component_correlatorSettings". The function checks the feasibility of the supplied settings and writes the values to the corresponding registers of the user app component.

The function "FMC110_user_app_component_correlatorEnable" enables the correlator application. All applications of the user app component are disabled using the function "FMC110_user_app_component_disableAllApps".

Using the API function "FMC110_writePatternToMemory", the memory interface is instructed to write data from the user app component to the DDR3 memory at a sequence of addresses. Arguments to that function are the start address, the sequence size and the write pattern selection as described in Section 4.8.

Likewise the function "FMC110_readSequenceFromMemory32" is used to read out a sequence of memory locations of the external DDR3 RAM, send it over the Ethernet and store the data to a buffer in the memory of the host computer. Using this API function the data is written to an array in LabVIEW which is afterwards inversely Fourier transformed and written to the hard drive of the host computer.

4.10 Monitoring and Fallback Reconfiguration

In order to prevent damage of the integrated circuits on the ML605 board and on the FMC110 mezzanine card it is beneficial to monitor the temperature and voltage of the components on these print circuit boards. Such a monitoring can also be useful to detect causes of erroneous measurements. In this section we describe the monitoring facilities we implemented in the firmware of the FPGA. Furthermore, a protection mechanism is presented that can prevent the FPGA from overheating. When an FPGA configuration is causing the temperature to rise above a certain threshold, a reconfiguration of the FPGA is initiated with a design that is known to have a low power consumption and thus should reduce the heating of the FPGA.

The temperature and supply voltages of the Virtex-6 FPGA are monitored by an on-chip temperature sensor and analog-to-digital converter which is represented by a primitive called system monitor (SYSMON). The system monitor is located in the center of the Virtex-6 die thus the measured temperature corresponds to the die temperature. The voltages which are measured are $V_{\mathrm{CC,INT}}$ and $V_{\mathrm{CC,AUX}}$. The voltage $V_{\mathrm{CC,INT}}$ is the supply voltage for the collectors (CC) of the transistors used to implement most of the internal logic of the FPGA. The voltage $V_{\mathrm{CC,INT}}$ has to be in the range from 0.95 V to 1.05 V as specified in [56]. The voltage $V_{\mathrm{CC,AUX}}$ which is specified to be in the range from 2.375 V to 2.625 V is another supply voltage for certain components of the Virtex-6 FPGA such as for the generation of the reference voltage for the differential inputs coming from the FMC110 mezzanine card.

The sysmon primitive can be configured through the Xilinx Core Generator software or by directly editing the VHDL module that has been generated by this tool which is called "sysmon_wiz_2_1" and is instantiated in the top-level module of the firmware ("wrapper_virtex6"). The SYSMON primitive allows the specification of threshold values for the temperature and voltages above which the system monitor issues an alert at a corresponding alert output port.

We implemented a module called "reconfiguration_with_watchdog" which contains a state machine that continuously resets the so called watchdog feature of the Virtex-6 [74]. The watchdog consists of a timer that automatically reconfigures the FPGA using a bit stream stored on the Numonyx flash memory of the ML605 board [55] if it is not reset within a certain time. For this feature to work, the jumpers and dip switches on the ML605 board have to be set correctly so that reconfiguration from the Numonyx flash memory is possible and the fallback reconfiguration has to be enabled in the options of the "bitgen" process that generates the programming file in the Xilinx ISE Design Suite. Furthermore the watchdog timer mode has to be set to "User" and a watchdog timer value has to be configured. Tests indicate that 2048 is a suitable value for the watchdog timer. Refer to the Virtex-6 configuration user guide [74] for more information about these settings.

Note that two different configurations can be stored on the Numonyx flash. The fallback reconfiguration is always done with the configuration that starts at address 0 of the flash. The "reconfiguration_with_watchdog" module can be used to manually initialize the reconfiguration and makes it possible to select one of the two configurations

stored on the Numonyx flash. We developed a small design where the top level module is called "wrapper_virtex6_fallback" which has a low power consumption. This design is also loaded when the ML605 board is switched on.

In addition to the temperatures and voltages monitored by the Virtex-6 system monitor, the ADT7411 monitor chip on the FMC110 card provides information on the temperature and voltages on the FMC110 mezzanine card. The ADT7411 chip can be controlled and read out over the I^2C bus which connects the FPGA to the monitor chip as mentioned in Section 4.2.

A state machine implemented on the FPGA continuously queries the measured temperature and voltages from the ADT7411 chip and stores the result into registers on the FPGA. This state machine is located in the VHDL module called "i2c_sysmon" which is a submodule of the "i2c_master" which is again a submodule of the "ml605_fmc110" module. The registers containing the measured temperature and voltages can be read out over the Ethernet using the API described in Section 4.9.

5 Measurement Setup

The principal ingredient of the experiments for which the measurement instrument presented in Section 4 is designed, is the physical implementation of a circuit QED system as discussed in Section 2. In order to control and probe this circuit QED system, signal generation equipment is needed. When a probe tone is sent to the input capacitor of the resonator, we are interested in the radiation emitted through the output capacitor. Thus a comprehensive signal analysis equipment is needed, part of which is formed by the FPGA-based measurement instrument developed in the present thesis. In the following we give an overview of the constellation of components used to perform the circuit QED experiment presented in Section 6.3.

A microscope image of the chip that comprises the superconducting circuit is shown in Figure 17. The chip comprises of two coplanar waveguide resonators. Each of the two resonators are capacitively coupled to a separate input waveguide. The output channels of the coplanar waveguide resonators are connected to a 50/50 beam splitter. The ground plane and center conductors of the waveguides are formed by a 150 nm thin niobium layer deposed on a sapphire waver.

The length of the coplanar waveguide resonators are 9.34 mm each. The center conductor has a width of 10 µm and is separated from the ground plane by a gap of 4.5 µm. The transmon qubits are placed in a small extent of the gap between center conductor and ground plane of each of the two coplanar waveguide resonators. The qubits are fabricated from aluminum with two Josephson junctions per qubit each consisting of a thin layer of aluminum oxide. Each qubit has a so called flux line attached to it. The flux lines are transmission lines which are shorted to the ground plane close to the qubits. A current running through these lines induces a magnetic flux through the loop formed by the two Josephson junctions of the transmon qubits. In that manner fast pulses can be applied to tune the Josephson energy E_J on a nano second timescale.

The theoretical considerations of Section 2 are only justifiable if the thermal population of the exited states of the quantum system under consideration is very low, i.e.

$$k_\mathrm{B}T \ll \hbar\omega_{01}, \tag{37}$$

where k_B is Boltzmann's constant, T is the temperature and ω_{01} is the transition energy from the ground state to the excited state. If this condition is satisfied, the system that consists of the resonator and the qubit will thermalize to its ground state as long as no external drive excites the system.

In order to fulfill the condition given by Equation (37), the chip is placed in a dilution refrigerator that cools the system down to a temperature of approximately 20 mK which is well below the critical temperature at which both niobium and aluminum becomes superconducting. The cryostat that has been used is the VeriCold Cryofree$^{\mathrm{TM}}$DR200-10 which uses a ^{3}He^{4}He mixture in a closed dilution circuit with a pulse tube cooler. The right part of Figure 18 shows the open cryostat with the different components of its interior as well as the different temperature stages. It can be seen from this figure that the cryostat is divided into several temperature stages where the temperature decreases from the top to the bottom. Each temperature stage is shielded against thermal radiation

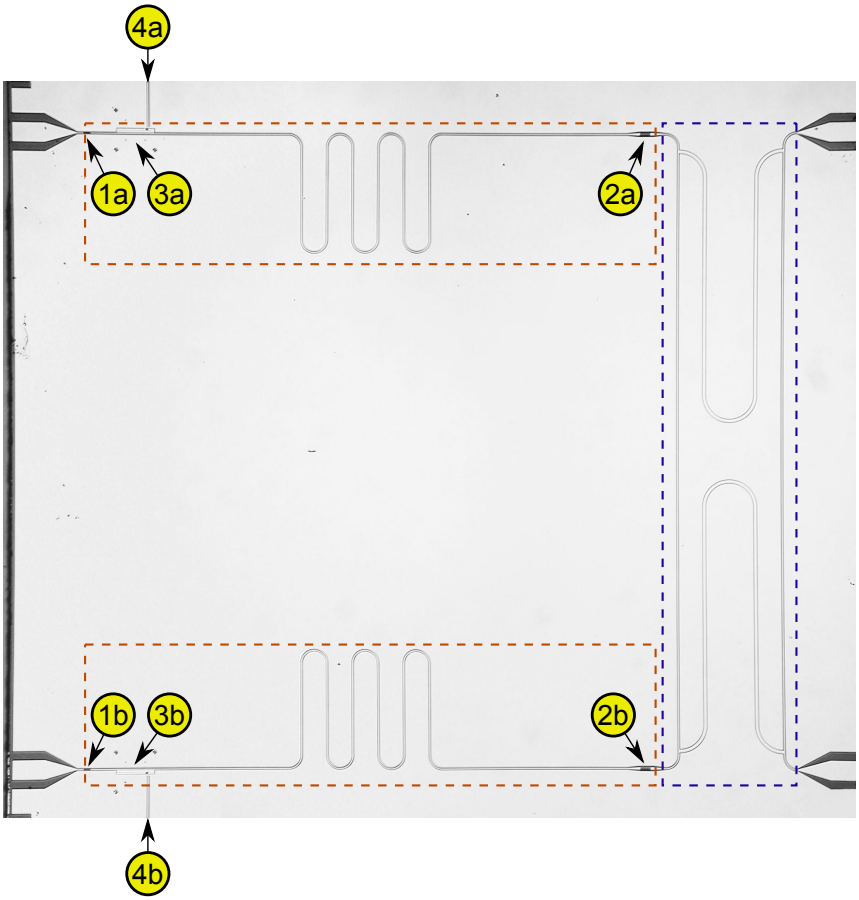


Figure 17: Grayscale optical microscope image of the superconducting circuit investigated in the present thesis. The two horizontal oriented dashed rectangles mark the two coplanar waveguide resonators. Labels **1a** and **1b** mark the input capacitor of resonators A and B respectively whereas labels **2a** and **2b** mark the output capacitors of the resonators. For resonator A and B, the labels **3a** and **3b** mark the extended gap where the respective transmon qubits A and B are located. The transmission lines **4a** and **4b** are used to apply flux pulses on the transmon qubits. The structure on the right of the image which is surrounded by the blue colored dashed rectangle implements a 50/50 beam splitter.

by a shield consisting of a metallic cylinder adjacent to the metallic round plates. Note that the round metal plates which are thermalized at the different temperature stages can be seen in Figure 18 whereas the shielding cylinders are added when the cryostat is closed and are thus not shown on this photograph. To prevent heat transfer from convection, a vacuum is produced in the chambers of the cryostat lowering the pressure to below 10^{-6}mbar.

The input signal of the resonators on the superconducting chip is generated at room temperature with the microwave generators which are denoted as "RF", "Alice" and "Stark" in the left part of Figure 18. The latter microwave generator labeled as "Stark" is used to generate the input signal for resonator A only, whereas the microwave generator labeled as "Alice" is used to generate the input signal for resonator B only. In contrast, the microwave generator "RF" is used to generate an input signal that reaches both resonators.

The microwave signal is then guided through semi-rigid coaxial cables with $50\,\Omega$ impedance to the cryostat. Because each charge carrier is subject to Johnson-Nyquist noise, i.e. noise induced by thermal agitation of the charge carriers, the generated input signal is attenuated between each thermal stage of the cryostat as can be seen from Figure 18. While passing the thermal stages upwards, the output lines run through circulators which prevent the conduction of Johnson-Nyquist noise in the direction towards the colder stages. The output signal is amplified at the 4 K stage by a linear high-electron-mobility transistor (HEMT) based amplifier.

For the signal analysis we use a heterodyne detection scheme in which the signal is first band-pass filtered and amplified before it is down-converted by mixing the signal with the microwave signal generated by the signal generator labeled as "LO" in Figure 18. The "I" output of the mixer is low-pass filtered suppressing frequency components higher than 400 MHz in order to prevent backfolding (aliasing) when the signal is sampled by the ADS5400 analog-to-digital converter (ADC0) on the FMC110 mezzanine card described in Section 4.2. This is because the sampling frequency of the analog-to-digital converter is $f_\mathrm{s} = 1\,\mathrm{GHz}$ which corresponds to a Nyquist frequency of 500 MHz.

Current sources were connected to the three superconducting coils below the sapphire waver in order to set the bias magnetic flux through the loop formed by the two Josephson junctions for each of the two transmon qubits. As described in Section 2.3, the resonance frequency of the transmon qubit depends on this magnetic flux.

The two flux lines which are shorted on the superconducting chip at the location of qubit A and B respectively are terminated by a $50\,\Omega$ resistor at room temperature since they are not used for the experiment described in Section 6.3.

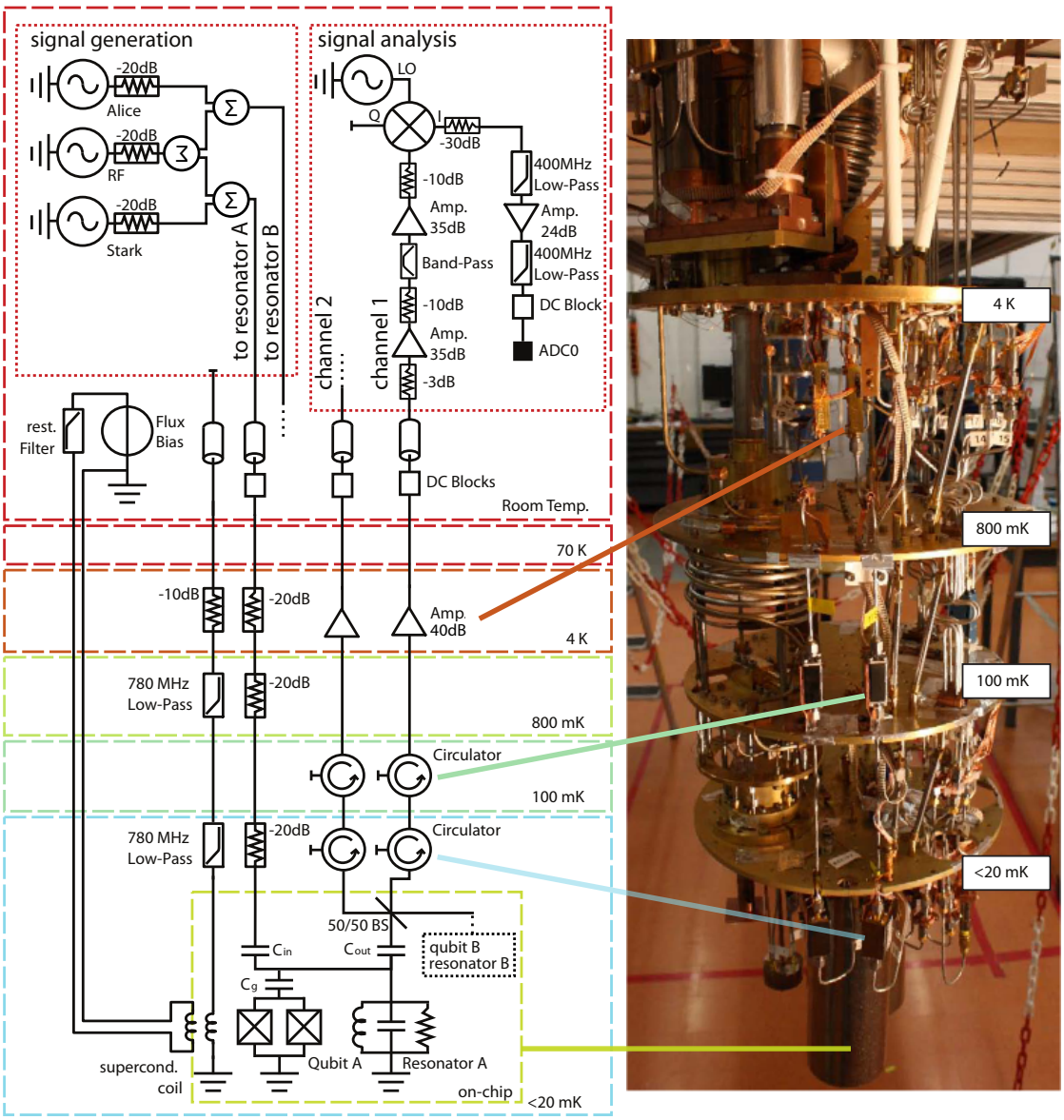


Figure 18: Setup of the cavity QED experiment. Left: the circuit that represents the measurement setup is shown. Right: photograph of the open dilution refrigerator showing different components and cabling at the different temperature stages.

6 Measurements and Characterizations

In order to test the suitability of the measurement instrument designed in the present thesis for circuit QED, several measurements have been performed which are presented in this chapter. In Section 6.1, the performance of the analog-to-digital converters on the FMC110 mezzanine card is characterized by the measurement of a test signal. In Section 6.2, an analog test signal generated with the digital-to-analog converters of the FMC110 is characterized using an oscilloscope and a spectral analyzer. The results of the circuit QED measurement performed to demonstrate the operation of the newly design measurement instrument are presented in Section 6.3.

6.1 Characterization of the analog-to-digital converter

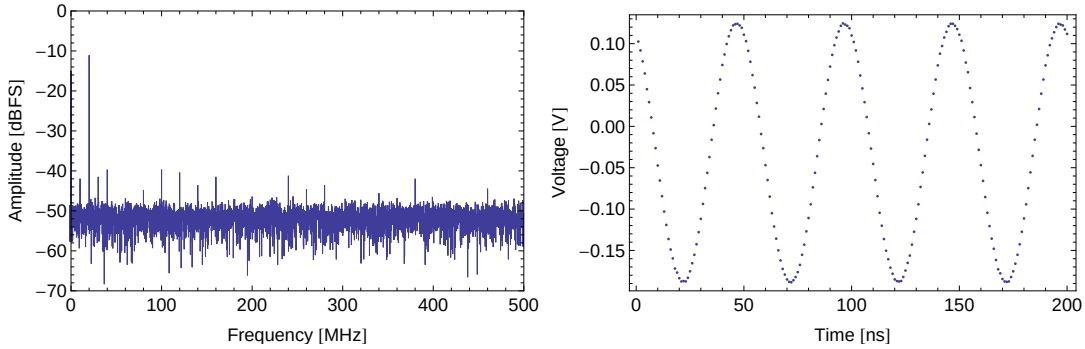
As discussed in Section 4.2, the FMC110 mezzanine board is equipped with two Texas Instruments ADS5400 12 bit analog-to-digital converters which can be operated at a maximal sampling frequency of 1 GHz. In the following we characterize the performance of the analog-to-digital converters by analyzing the raw data that has been sampled from an analog signal generated by an Agilent 33220A function / arbitrary waveform generator.

For the characterization, the function generator has been configured to generate a sine wave with a frequency of 20 MHz and an amplitude of $500\,\mathrm{mV_{pp}}$ with respect to a $50\,\Omega$ load. The generated signal was split with a BNC splitter to 2 coaxial cables with $50\,\Omega$ impedance each. One coaxial cable was connected to the analog-to-digital converter referred to as ADC0. The second coaxial cable was connected to the other analog-to-digital converter referred to as ADC1.

We sampled 8192 data points by operating each of the analog-to-digital converters at the full frequency of 1 GHz with a configured full-scale voltage of $2\,\mathrm{V_{pp}}$. The reference firmware buffers the data in a first-in-first-out (FIFO) memory on the FPGA. This FIFO was read out over the Ethernet by a host computer. To obtain good statistics, the measurement has been repeated 100 times. Note that for ADC1, 20 of the 100 recorded sequences contained erroneous samples (outliers) which can be explained to be the result of synchronization errors between the FPGA and the analog-to-digital converters. In the firmware we developed to perform the measurement presented in Section 6.3, we improved the synchronization routine as discussed in Section 4.6.2. Figures 19b and 19d show the first 200 samples of one particular time trace which contained no samples that can be identified as outliers. On the host computer, the first 8150 samples were Fourier transformed to obtain the corresponding spectral density shown in Figures 19a and 19c.

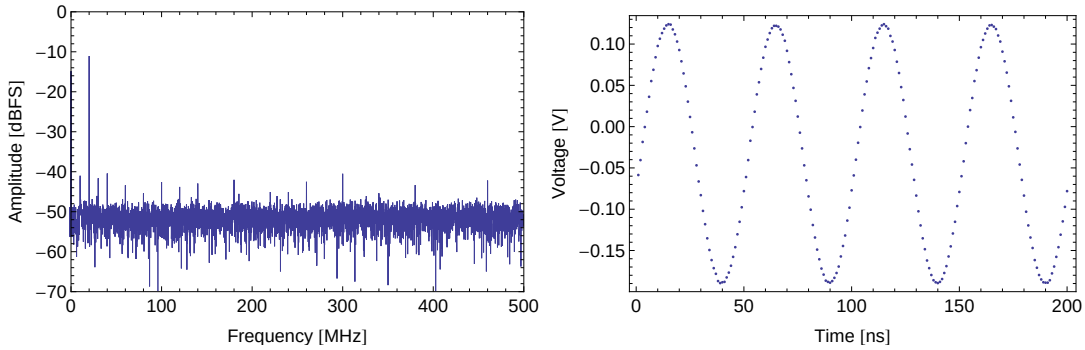
The time traces shown in Figures 19b and 19d are not phase aligned as the measurement was not synchronized by a signal dependent trigger and the measurements with ADC0 and ADC1 were not done simultaneously.

Due to the splitting of the signal to both analog-to-digital converters we assume that each converter receives one half of the total power of the signal. Therefore we expect that the amplitude at the input of each analog-to-digital converter corresponds to $500\,\mathrm{mV_{pp}}/\sqrt{2} \approx 354\,\mathrm{mV_{pp}}$. By averaging over the Fourier transform of 80 sequences



(a) ADC0 spectral denisty

(b) ADC0 time trace



(c) ADC1 spectral density

(d) ADC1 time trace

Figure 19: Characterization of the analog-to-digital converters by recording a sine wave signal originated from a function generator. Parts (a) and (c) show the Fourier transform of a sequence of 8150 samples obtained from ADC0 and ADC1 respectively. Parts (b) and (d) show the corresponding first 200 samples of the sequence in the time domain.

that contain no outliers we obtain that the average amplitude of the signal in the frequency bin centered at 20 MHz has a peak-to-peak voltage of 313 $\mathrm{mV_{pp}}$ on ADC0 and 313 $\mathrm{mV_{pp}}$ on ADC1 with a standard deviation of 77 $\mu\mathrm{V_{pp}}$ and 72 $\mu\mathrm{V_{pp}}$ respectively.

It can also be seen from the Figures 19b and 19d that the measured signal suffers from a significant direct current (DC) offset of $-(31.84 \pm 0.02)$ mV on ADC0 and $-(32.84 \pm 0.03)$ mV on ADC1. Besides the fact that the observed DC offset reduces the dynamic range for negative voltages, the DC offset usually does not play a significant role in circuit QED experiments unless both quadratures of an analog down converted signal are sampled by the two analog-to-digital converters so that the difference in the DC offsets of ADC0 and ADC1 is relevant.

A quantity which is of common use to characterize the performance of analog-to-digital converters is the so called signal-to-noise and distortion (SINAD) which is defined as [58]

$$\mathrm{SINAD} \equiv 10 \log_{10} \frac{P_\mathrm{S}}{P_\mathrm{N} + P_\mathrm{D}}, \tag{38}$$

where P_S is the power of the expected signal to the power of the noise P_N and the power of the harmonic distortion P_D, i.e. the power in bins that are located at multiples of the signal frequency. The sum $P_\mathrm{N} + P_\mathrm{D}$ is thus equal to the sum of the power in each frequency bin of the spectrum excluding the DC and the signal bins. From the Fourier transform of the measured time sequences a SINAD value of (42.4 ± 0.07) dBc for ADC0 and a value of (42.2 ± 0.20) dBc for ADC1 can be calculated.

The SINAD is used to define an effective number of bits (ENOB) that contain non-random information by the formula [58]

$$\mathrm{ENOB} \equiv \frac{\mathrm{SINAD} - 1.76}{6.02}. \tag{39}$$

By this formula, the ENOB that correspond to the measured SINAD of ADC0 and ADC1 are 6.75 and 6.71 respectively.

It is common [75] to extrapolate the SINAD to the full-scale voltage by replacing P_S in the numerator of Formula (38) with the square of the power that corresponds to the full-scale voltage. Doing this leads to a full-scale ENOB of 9.43 for ADC0 and 9.39 for ADC1. However this extrapolation is based on the assumption that the power of the harmonic distortion does not significantly scale with the power of the signal. This assumption is likely to be invalid to some degree. Nevertheless the full-scale values for ENOB measured here are in good agreement with the values specified in the data sheet of the ADS5400 [58].

6.2 Characterization of the digital-to-analog converter

One possible future application of the FPGA is to send feedback signals to the circuit QED chip as theoretically investigated in [13]. Moreover, for quantum algorithms that involve feedback such as teleportation [14], the field programmable gate array could become an important part in the feedback control loop as it could possibly be used to

generate e.g. flux pulses based on a Bell measurement of a two-qubit state in order to rotate the state vector of the target qubit into the desired state.

The generation of flux pulses as well as shaped microwave pulses requires the generation of an analog signal. As seen in Section 4.2, the FMC110 mezzanine card features two Texas Instruments DAC5681Z 16-bit digital-to-analog converters capable of converting 1 billion digital data points per second to analog voltages with a specified full-scale voltage of $V_{\mathrm{fs}} = 1.0\,\mathrm{V_{pp}}$ with respect to a 50Ω load. In the following, the analog output signal of these digital-to-analog converters is characterized.

For the analysis, the digital representation of a 62.5 MHz sine wave with an amplitude corresponding to the full-scale voltage has been transferred from the host computer to a waveform memory on the FPGA which is then continuously sent to the digital-to-analog converter using the original firmware by 4DSP. In the firmware, these digital-to-analog converters are referred to as DAC0 and DAC1. We adapt this naming convention for the presentation of the following results.

The effective peak-to-peak voltage was measured with a LeCroy SDA 13000 oscilloscope. We measured a peak-to-peak voltage of $(900 \pm 6)\ \mathrm{mV_{pp}}$ for the output signal of DAC0 and $(901 \pm 6)\ \mathrm{mV_{pp}}$ for DAC1.

Furthermore, the total harmonic distortion was measured, which is the ratio of the sum of the powers of all harmonic components of the signal to the signal power itself. We did this by measuring the analog output of DAC0 with the Agilent E4407B spectrum analyzer.

For the full-scale sine wave, the power of the signal from DAC0 measured with the spectrum analyzer was $P_{\mathrm{signal}} \approx 1.52\,\mathrm{dBm}$ whereas the total harmonic distortion THD taking into accout the first 10 harmonic components of the signal was measured to be $\mathrm{THD} \approx -68.5\,\mathrm{dBc}$. Figure 20 shows the measured power spectrum showing the first 3 harmonics.

We then reduced the amplitude of the digital sine wave to $1/4V_{\mathrm{fs}}$ and repeated the measurements with the spectrum analyzer. For this situation we measured a signal power of $P_{\mathrm{signal}} \approx -11.3\,\mathrm{dBm}$ and total harmonic distortion of $\mathrm{THD} \approx -65.58\,\mathrm{dBc}$

6.3 Mollow Triplets in a Circuit QED Experiment

In order to demonstrate the main benefit of the newly designed FPGA-based measurement instrument for circuit QED experiments, we measured the resonance fluorescence of the coupled atom-cavity system in the circuit QED setup described in Chapter 5. In the present section, these results are presented and compared to the theoretical predictions described in Section 2.7.

For the measurement presented in this section only qubit A and resonator A of the superconducting circuit described in Chapter 5 is relevant since no electromagnetic drive has been applied to resonator B.

During the entire measurement procedure, the output channel 1 was connected to ADC0 of the Virtex-6 FPGA-based measurement instrument which is developed in the present thesis and output channel 2 was connected to the previous Virtex-4 FPGA board.

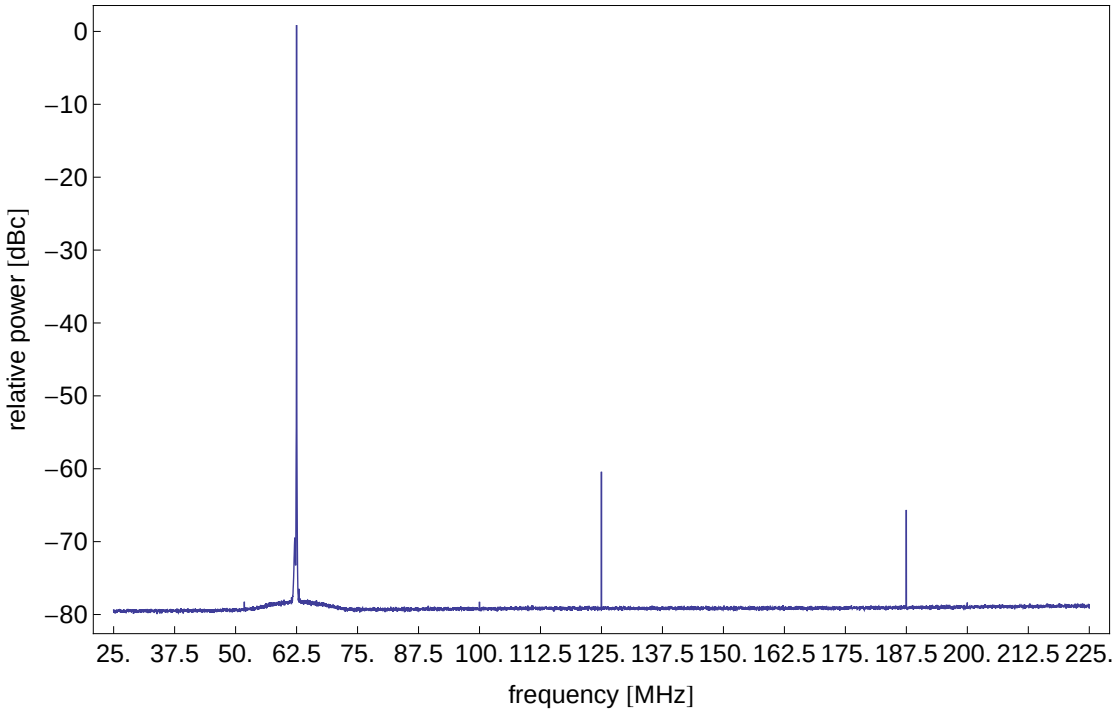


Figure 20: Power spectrum of the output of the digital-to-analog converter DAC0. A sine wave of 62.5 MHz is converted from digital to analog with a specified peak-to-peak voltage of 1 V_{pp}.

The first step is the determination of the magnetic flux ϕ through the loop formed by the Josephson junctions of the transmon qubit needed to tune the transition frequency between ground and first excited state of the transmon into resonance with the fundamental mode of the coplanar waveguide resonator. For this a probe tone with frequency ν_{probe} and power of $P_{\text{input}} = -35\,\text{dBm}$ is sent from the input microwave generator (Stark) to the input of the coplanar waveguide resonator. With this tone at the input, the power of the coherently transmitted signal in the frequency bin centered at ν_{probe} was measured using the previous Virtex-4 FPGA board sampling at a frequency of 100 MHz. While repeating this experiment, the magnetic flux ϕ was varied from $0.125\phi_0$ to $0.175\phi_0$ in steps of $0.005\phi_0$ as well as over the probe frequency ν_{probe} from 6.4 GHz to 6.9 GHz in steps of 400 kHz in a two-dimensional sweep. In each step, the time trace consisting of 1024 samples was averaged over 4096 successive measurements.

The magnetic flux at which the qubit is in resonance with the resonator mode is found by searching the avoided crossing of the energy levels that correspond to the first two excited states of the combined system of resonator and qubit. The avoided crossing is at the point where the energy difference between these two levels is minimized as shown in Figure 21. As argued in Section 2.5, the difference between the energy levels at this avoided crossing corresponds to the vacuum Rabi mode splitting of $2g$ from which we determined that the coupling constant has a value of $g/2\pi \approx 148\,\text{MHz}$ in our system. Furthermore we determined the resonator mode frequency to be $\omega_{\text{r}}/2\pi \approx 6.6559\,\text{GHz}$ by determining the center frequency of the vacuum Rabi mode splitting.

Before measuring the resonance fluorescence power spectrum, another spectroscopic measurement has been performed to determine the transition frequency $\omega_{1-} = \omega_{\text{r}} - g$ between ground and excited state of the combined system more accurately since 72 hours passed since the calibration measurement described above has been performed. During this calibration measurement, the qubit was tuned into resonance with the resonator mode by setting the magnetic flux ϕ to the value which tunes the qubit into resonance with the fundamental mode as determined in the first calibration measurement. The power of the input microwave generator (Stark) has been set to $P_{\text{input}} = -35\,\text{dBm}$. The frequency ν_{probe} of the input signal was varied from 6.46 GHz to 6.56 GHz in steps of 200 kHz. By measuring the coherently transmitted signal using the previous FPGA in the same manner as described above and fitting a Lorentzian function to the obtained transmission spectrum, we determined a value of $\omega_{1-}/2\pi \approx 6.51112\,\text{GHz}$ for the transition frequency between the ground and the first excited state of the resonator-qubit system.

The resonance-fluorescence measurement was performed with a bandwidth of 500 MHz which is made accessible by the combination of the ADS5400 analog-to-digital converters and the Virtex-6 FPGA with the firmware described in Section 4.3. As for the second calibration measurement, the qubit was tuned to resonance with the fundamental mode of the resonator by applying the magnetic flux determined in the first measurement.

The microwave generator (Stark) at the resonator input has been set to $\nu_{\text{drive}} = 6.51112\,\text{GHz}$ which corresponds to the measured value of ω_{1-}. The frequency of the LO signal generator used for the analog down-conversion of the output signal was set to $\nu_{\text{LO}} = \nu_{\text{drive}} - 125\,\text{MHz}$. With this setting, the coherently scattered signal is expected

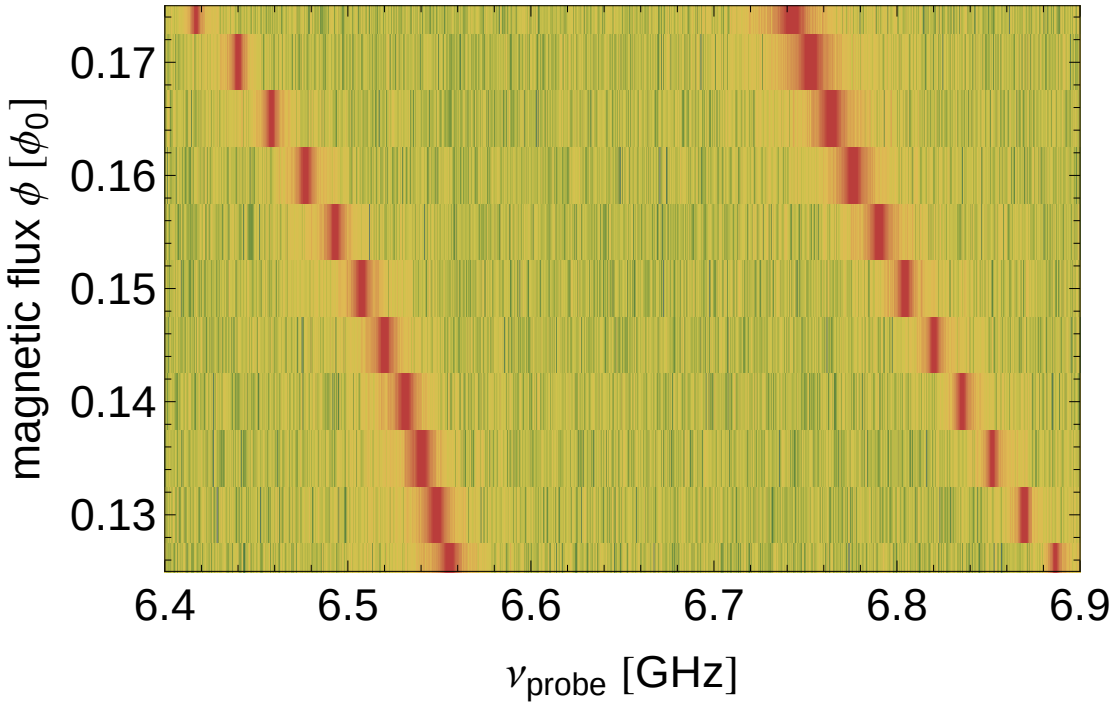


Figure 21: The colors indicate the received logarithmic power of a probe tone that has been transmitted through the resonator-qubit system as a function of the probe frequency ν_{probe} and the magnetic flux ϕ through the loop formed by the two Josephson junctions of the transmon qubit. The flux is given in units of the flux quantum ϕ_0.

at a frequency of $\nu_{\text{carrier}} = 125\,\text{MHz}$.

Using the ADS5400 (ADC0), we sampled 8192 data points of the filtered signal at a frequency of 1 GHz at each measurement trigger. The trigger repetition period was set to 16 µs. The firmware of the Virtex-6 FPGA was configured to Fourier transform the time trace of 8192 data points after every trigger reception and average the obtained absolute square values. For background subtraction, the input microwave generator was switched on and off between every measurement trace. Correspondingly, each measured power spectrum contributed to the average with an alternating sign. After 32 million trigger repetitions, the averaged power spectrum was temporarily stored in the external DDR3 RAM of the Virtex-6 FPGA board before it was transferred to the host computer and stored on the hard drive by the LabVIEW-based measurement control software.

The LabVIEW software was used to sweep the power P_{input} of the input signal generator (Stark) from $-21\,\text{dBm}$ to $+9\,\text{dBm}$ in steps of 3 dBm. In every step of the sweep, the above described FPGA-based measurement has been repeated. In this way we obtained the 11 power spectra shown in Figure 22.

We simultaneously fitted the theoretical function for the Mollow triplet spectrum of a driven two level system [22] to the data obtained for all microwave generator powers P_{input} smaller or equal to 0 dBm. For the 3 power spectra that were obtained when the microwave generator was set to $P_{\text{input}} > 0\,\text{dBm}$, we extrapolated the parameter values based on the fitted parameters we obtained for the lower powers.

While the fit to the theoretical model seems to be valid for relatively low driving amplitudes which correspond to Rabi frequencies below $\Omega_{\text{R}} \approx 40\,\text{MHz}$, it can be clearly seen that the data is not in agreement with the theoretical prediction when the separation between the satellite peaks and the center peak becomes larger than 40 MHz. It is also noticeable that the height of the center peak is reduced for large drive strengths. These observations may indicate new physics which can be investigated further with the new FPGA-based measurement instrument.

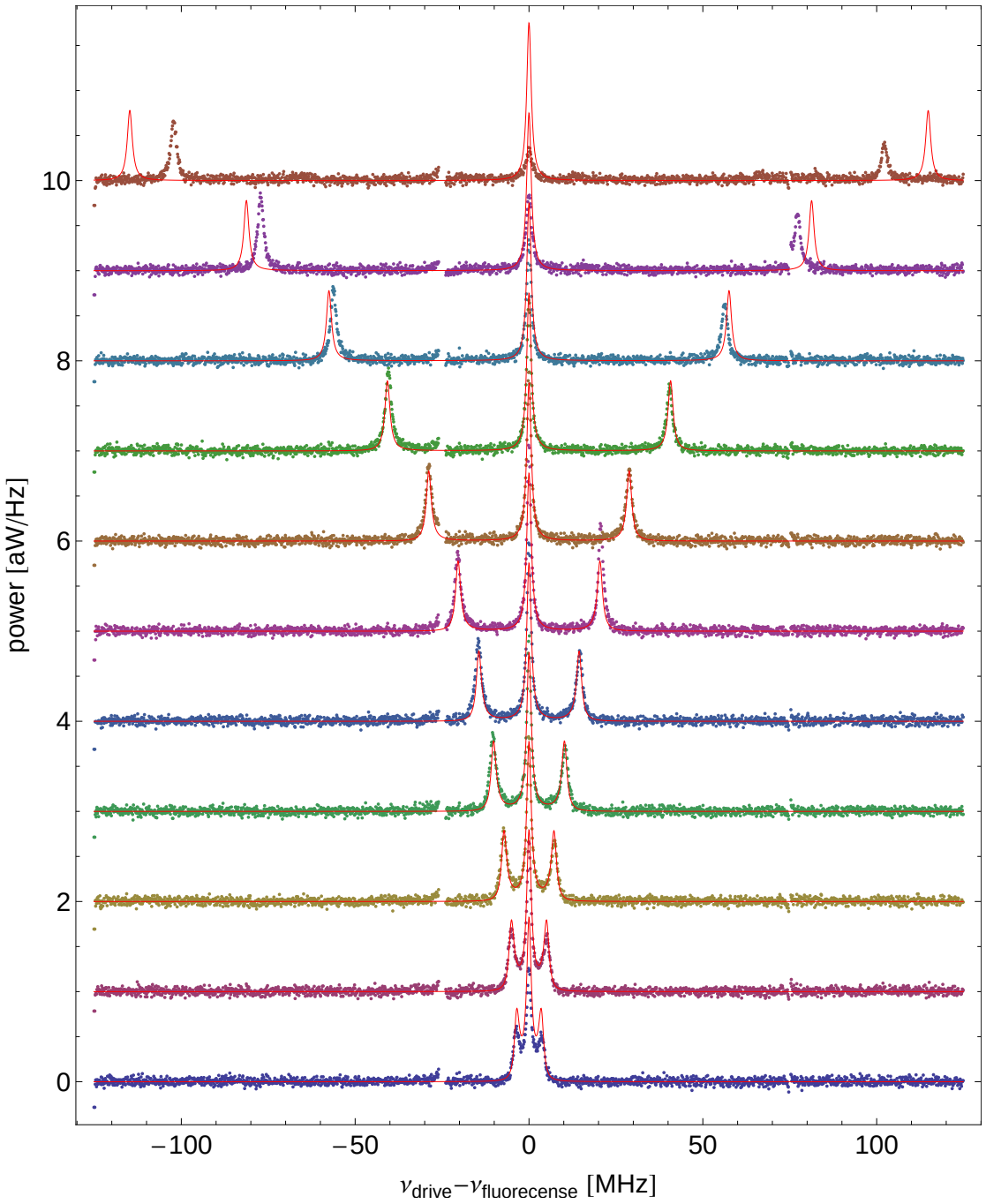


Figure 22: Power spectra of the resonance fluorescence coming from a transmon qubit coupled to a coplanar waveguide resonator. Between each power spectrum from the bottom to the top, the drive power P_{input} of the microwave generator (Stark) is increased in steps of 3 dBm. The continuous red curves represent a fit to the theoretical model.

7 Conclusions

A new measurement instrument for circuit QED measurement setups based on a field programmable gate array (FPGA) has been developed capable of performing correlation function measurements at a bandwidth of 500 MHz.

A large part of this development task consisted in the design and implementation of a firmware for the FPGA which can process the data at a rate of 1 billion samples per second. The firmware is designed to be extensible for future requirements.

We implemented a fast Fourier transform (FFT) algorithm in a parallelized manner to be able to process the data in real-time. Using the FFT, the first-order correlation function is computed in the frequency domain on the FPGA by averaging over the absolute square of the Fourier coefficients.

A memory interface has been designed to temporarily store the data on an external random-access memory (RAM) in order to extend the memory available on the FPGA.

We demonstrated the instrument by measuring the power spectrum of resonance fluorescence coming from a superconducting qubit coupled to a coplanar waveguide resonator. In this measurement, we observed a Mollow-triplet like spectrum with a maximal separation between the center peak and each of the satellite peaks of 102 MHz.

The fit to the theoretical prediction for a two-level system is possible for Rabi frequencies below 40 MHz but cannot be used to extrapolate the spectrum for higher drive amplitudes. This indicates new physics which have yet to be theoretically explained as well as further examined experimentally using the high-bandwidth measurement instrument developed in the present thesis.

Acknowledgments

My primary acknowledgment goes to Professor Dr. Andreas Wallraff for offering me the interesting design and research task and for the opportunity to work in his lab.

Many thanks go to my adviser Christian Lang who has helped me a lot in the process of getting familiar with FPGA programming and our measurement setup, answered many practical and theoretical questions and directed the work towards its goal. He has also sparked my interest in the possibility to design measurement instruments based on FPGA in the beginning when I was deciding on a topic for the present master thesis.

I also had helpful discussions about the theory of circuit QED with Chrsitopher Eichler, Marek Pechal, Tobias Thiele, Lars Steffen and Dr. Abdufarrukh Abdumalikov.

Everyone in the Qudev-Team was ready to help basically at any time when I had a question about one of our measurement instruments for which I am very grateful.

Last but not least I want to thank my life partner Claudia Born for her continuous emotional support.

References

[1] Wallraff, A. et al., Nature **431** (2004) 162.

[2] Walther, H., Varcoe, B. T. H., Englert, B.-G., and Becker, T., Reports on Progress in Physics **69** (2006) 1325.

[3] Houck, A. et al., Nature **449** (2007) 328.

[4] Bozyigit, D. et al., J. Phys.: Conf. Ser. **264** (2011) 012024.

[5] Bozyigit, D. et al., Nat. Phys. **7** (2011) 154.

[6] Lang, C. et al., Phys. Rev. Lett. **106** (2011) 243601.

[7] Blais, A., Huang, R.-S., Wallraff, A., Girvin, S. M., and Schoelkopf, R. J., Phys. Rev. A **69** (2004) 062320.

[8] Blais, A. et al., Phys. Rev. A **75** (2007) 032329.

[9] Majer, J. et al., Nature **449** (2007) 443.

[10] Fink, J. M. et al., Collective qubit states and the tavis-cummings model in circuit qed, 2008.

[11] DiCarlo, L. et al., Nature **460** (2009) 240.

[12] Fedorov, A., Steffen, L., Baur, M., da Silva, M. P., and Wallraff, A., Nature **advance online publication** (2011) .

[13] Liu, Z. et al., Phys. Rev. A **82** (2010) 032335.

[14] Bennett, C. H. et al., Phys. Rev. Lett. **70** (1993) 1895.

[15] Tada, M. et al., Physics Letters A **349** (2006) 488.

[16] Jones, M. L., Wilkes, G. J., and Varcoe, B. T. H., Single microwave photon detection in the micromaser, arXiv:0905.0166v1 [quant-ph], 2009.

[17] Johnson, B. R. et al., Nat. Phys. (2010) 663 .

[18] Chen, Y. F. et al., Microwave photon counter based on josephson junctions, 2010.

[19] Miki, S. et al., Appl. Phys. Lett. **99** (2011) 111108.

[20] Glauber, R. J., Phys. Rev. **130** (1963) 2529.

[21] Glauber, R. J., Phys. Rev. Lett. **10** (1963) 84.

[22] Mollow, B. R., Phys. Rev. **188** (1969) 1969.

[23] Lyons, R. G., *Understanding Digital Signal Processing (2nd Edition)*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.

[24] Yamamoto, Y. and Imamoğlu, A., *Mesoscopic Quantum Optics*, Wiley, 1999.

[25] Büttiker, M., Phys. Rev. B **36** (1987) 3548.

[26] Bouchiat, V., Vion, D., Joyez, P., Esteve, D., and Devoret, M. H., Phys. Scr. **T76** (1998) 165.

[27] Josephson, B. D., Physics Letters **1** (1962) 251.

[28] Tinkham, M., *Introduction to Superconductivity*, McGraw-Hill International Editions, 1996.

[29] Girvin, S. M., Superconducting qubits and circuits: Artificial atoms coupled to microwave photons, Lectures delivered at Ecole d'Eté Les Houches; To be published by Oxford University Press, 2011.

[30] Schuster, D. I., *Circuit Quantum Electrodynamics*, PhD thesis, Yale University, 2007.

[31] Cottet, A., *Implementation of a quantum bit in a superconducting circuit*, PhD thesis, Universite Paris 6, 2002.

[32] Koch, J. et al., Phys. Rev. A **76** (2007) 042319.

[33] Göppl, M. et al., J. Appl. Phys. **104** (2008) 113904.

[34] Wallraff, A., Physik Journal **7** (2008) 39.

[35] Schuster, I. et al., Nat. Phys. **4** (2008) 382.

[36] Romero, G., García-Ripoll, J. J., and Solano, E., Phys. Rev. Lett. **102** (2009) 173602.

[37] Tian, L. and Carmichael, H. J., Phys. Rev. A **46** (1992) R6801.

[38] Imamoğlu, A., Schmidt, H., Woods, G., and Deutsch, M., Phys. Rev. Lett. **79** (1997) 1467.

[39] Carmichael, H. J., *Statistical Methods in Quantum Optics 1: Master Equations and Fokker-Planck Equations*, Springer-Verlag, 1999.

[40] Bishop, L. S. et al., Nat. Phys. **5** (2009) 105.

[41] Sabah, S. and Lorenz, R., Design and calibration of IQ-mixers, in *6th European Particle Accelerator Conference, Stockholm, Sweden*, 1998.

[42] Xilinx, Inc., *Virtex-6 FPGA Configurable Logic Block*, 1.1 edition, 2009.

[43] Yalamanchili, S., *Introductory VHDL: From Simulation To Synthesis*, Prentice Hall, 2001.

[44] Fernandez, M. and Abusaidi, P., Virtex-6 FPGA routing optimization design techniques, Technical report, Xilinx, Inc., 2010.

[45] Xilinx, Inc., *Virtex-6 FPGA DSP48E1 Slice User Guide*, 1.3 edition, 2011.

[46] Xilinx, Inc., *Virtex-6 FPGA Memory Resources User Guide*, 1.6 edition, 2011.

[47] Xilinx, Inc., *Virtex-6 Family Overview*, 2.2 edition, 2010.

[48] Hoppe, B., *Verilog Modellbildung für Synthese und Verifikation*, Oldenbourg Verlag München Wien, 2006.

[49] Xilinx, Inc., *System Generator for DSP User Guide*, 13.2 edition, 2011.

[50] Bozyigit, D., Design and development of an high performance signal processing platform for qubit readout, Technical report, Laboratory of solid state physics, ETH Zurich, 2008.

[51] Lang, C., Read-out strategies for multi-qubit states in circuit quantum electrodynamics, Diploma thesis, LMU Munich, 2009.

[52] Eichler, C. et al., Phys. Rev. Lett. **106** (2011) 220503.

[53] Nallatech Limited, *XtremeDSP Development Kit-IV User Guide*, 2005.

[54] Seelam, R., I/O design flexibility with the FPGA mezzanine card (FMC), Technical report, Xilinx, Inc., 2009.

[55] Xilinx, Inc., *ML605 Hardware User Guide*, 1.5 edition, 2011.

[56] Xilinx, Inc., *Virtex-6 FPGA Data Sheet: DC and Switching Characteristics*, 3.1 edition, 2011.

[57] 4DSP LLC, USA, *FMC110 User Manual*, 1.9 edition, 2010.

[58] Texas Instruments Incorporated, *ADS5400 12-Bit, 1-GSPS Analog-to-Digital Converter Data Sheet*, 2010.

[59] Texas Instruments Inc., *16-BIT, 1.0 GSPS 2x-4x Interpolating Digital-To-Analog Converter (DAC).*

[60] Analog Devices, Inc., *12-Output Clock Generator with Integrated 2.0 GHz VCO*, 2011.

[61] Analog Devices, Inc., *SPI-/I2C-Compatible, 10-Bit Digital Temperature Sensor and 8-Channel ADC*, 2006.

[62] 4DSP LLC, 955 S Virginia Street, Suite 214, Reno, NV 89502, USA., *FMC110 Reference Firmware For ML605*, 1.0 edition, 2010.

[63] Wiltgen, J. and Ayer, J., Bus master DMA performance demonstration reference design for the Xilinx endpoint PCI Express® solutions (XAPP1052), Technical report, Xilinx, Inc., 2010.

[64] Gaschet, C., Xilinx Xcell **35** (2000) 39.

[65] Chaney, T. and Molnar, C., Computers, IEEE Transactions on **C-22** (1973) 421 .

[66] Alfke, P., Metastable recovery in Virtex-II Pro FPGAs (XAPP094), Technical report, Xilinx, Inc., 2005.

[67] Cadence Design Systems, Inc., *Clock Domain Crossing, Closing the Loop On Clock Domain Functional Implementation Problems*, 2004.

[68] Xilinx, Inc., *Virtex-6 FPGA Memory Interface Solutions (DS186)*, 2011.

[69] Xilinx, Inc., *Virtex-6 FPGA SelectIO Resources User Guide*, 2010.

[70] Day, B., 16-channel, DDR LVDS interface with real-time window monitoring, Technical report, Xilinx, Inc., 2008.

[71] Riccardi, D. and Novellini, P., An attribute-programmable PRBS generator and checker, Technical report, Xilinx, Inc., 2011.

[72] Defossez, M., Connecting Virtex-6 FPGAs to ADCs with serial LVDS interfaces and DACs with parallel LVDS interfaces, Technical report, Xilinx, Inc., 2010.

[73] Cooley, J. W. and Tukey, J. W., Mathematics of Computation **19** (1965) 297.

[74] Xilinx, Inc., *Virtex-6 FPGA Configuration User Guide*, 2010.

[75] Kester, W., Understand SINAD, ENOB, SNR, THD, THD + N, and SFDR so you don't get lost in the noise floor, Technical report, Analog Devices, Inc., 2008.