Noname manuscript No.
(will be inserted by the editor)

Synchronous Sampling and Clock Recovery of Internal Oscillators for Side Channel Analysis and Fault Injection

Colin O'Flynn · Zhizhang (David) Chen

Received: date / Accepted: date

Abstract Measuring power consumption for side-channel analysis typically uses an oscilloscope, which measures the data relative to an internal sample clock. By synchronizing the sampling clock to the clock of the target device, the sample rate requirements are considerably relaxed; the attack will succeed with a much lower sample rate.

This work measures the performance of a synchronous sampling system attacking a modern microcontroller running a software AES implementation. This attack is characterized under four conditions: with a stable crystal-oscillator based clock, with a clock that is randomly varied between 3.9 MHz–13 MHz, with an internal oscillator that is randomly varied between 7.2 MHz–8.1 MHz, and with an internal oscillator that has slight random variation due to natural 'drift' in the oscillator.

Traces captured with the synchronous sampling technique can be processed with a standard Differential Power Analysis (DPA) style attack in all four cases, whereas when an oscilloscope is used only the stable oscillator setup is successful. This work also develops the hardware to recover the internal clock of a device which does not have an externally available clock. It is possible to implement this scheme in software only, allowing it to work with existing oscilloscope-based test environments.

Keywords side-channel analysis, acquisition, synchronization, DPA

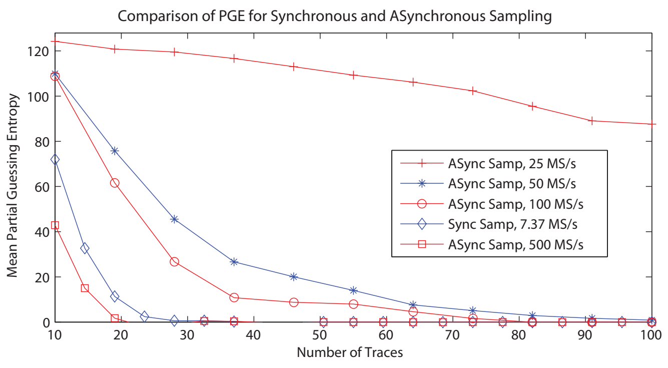


Fig. 1 The synchronous sampling method used here simplifies measurement work by allowing the use of a slower sampling speed, and thus shorter traces, with similar success rates to a high-speed oscilloscope.

1 Introduction

By measuring the power consumed by a digital device on each clock cycle, it is possible to infer something about the data being processed by this device. This was demonstrated as a method of breaking cryptographic cores using Differential Power Analysis (DPA)[9]. Such measurements are typically done with standard oscilloscopes, which depending on the attack algorithm and device under attack may range from simple low-cost oscilloscopes to high-end specialist oscilloscopes. But if the underlying objective is to measure data on the clock edges of the system clock, sampling at the clock rate of the system is sufficient, provided such samples occur at the correct moment (i.e. on the clock edge). This sampling technique is called synchronous sampling, where the sample clock is synchronized to the device clock.

Dalhousie University, Halifax, Canada
E-mail: {coflynn, z.chen}@dal.ca

The application of this to side-channel analysis was described and demonstrated in [12], where the SASEBO-GII board was attacked, and this demonstrated how sampling at 96 MS/s synchronously achieved similar results to sampling at 2 GS/s asynchronously.

For this to be successful, the previous work assumed that the system clock was readily available. For many systems this will be the case—an external oscillator or clock drives the digital logic, and it is trivial to tap into this clock. But many devices rely instead on an internal oscillator; there is no clock signal available for synchronous sampling. In addition devices may purposely vary the frequency of the internal oscillator in an attempt to stop power traces from synchronizing in the time domain, requiring the attacker to resynchronize the traces after capture. The varying clock countermeasure is assumed to be difficult to reverse in most instances. For example it is claimed in [18] that varying the clock frequency "makes time correlation, a very important step in power analysis attacks, impossible.". Beyond side-channel analysis, the generation of a synchronized clock can be used for injecting faults or glitches on specific clock cycles.

This work addresses these two issues. First, an introduction to the reference platform being used is given, along with a comparison of the synchronous sampling technique to standard asynchronous sampling on this platform. The platform is then changed to use an internal oscillator which actively varies the frequency during cryptographic operations, and the performance of synchronous sampling is demonstrated. Finally a method of performing clock recovery, and using that clock for synchronous sampling is demonstrated. The clock recovery method can also be performed in software, allowing the use of 'synchronous sampling' on existing oscilloscope-based platforms.

2 Experimental Platform

The device under test (DUT) is an Atmel AtMega48A microcontroller in 28-pin DIP. This device was selected due to several clocking features: it can use an internal or external clock source, the internal oscillator can be adjusted by firmware running on the microcontroller during operation, and the internal clock can be output onto an I/O pin. The differential voltage is measured across a shunt inserted into the VCC pin of the microcontroller. For asynchronous sampling an Agilent MSO 54831D oscilloscope is used, and for synchronous sampling the OpenADC is used along with a ZTEX FPGA board. Full details of the capture hardware and software are available in [13] and at the ChipWhisperer wiki[1]. See Fig. 13 for a photo of the test setup.

The 'A' suffix for the AtMega48A indicates it is using a recent fabrication process; the older AtMega48P by comparison is made with a larger (0.35μm) process. The AtMega48P draws more power, and thus would be expected to give a stronger signal across the resistive shunt used to measure current. The AtMega48A thus reflects a reasonable platform which can be compared against any recent digital IC.

The crypto module under attack is a C implementation of the AES-128 algorithm. The specific C implementation chosen was 'AES in C' available from avrcryptolib[2]. The attack algorithm is a standard Correlation Power Analysis (CPA) attack[3].

A comparison of asynchronous sampling and synchronous sampling is shown in Fig. 1. For this figure an external 7.37 MHz crystal oscillator was used as a clock source. Results in this paper will be an average of the partial guessing entropy (PGE) of all subkeys, and where space permits the PGE of each individual subkey is graphed. A brief overview of the PGE metric is provided in the following section.

2.1 Meaning of PGE

The 'guessing entropy' can be defined as the "average number of successive guesses required with an optimum strategy to determine the true value of a random variable X"[10]. In this paper the 'optimum strategy' is to take the output of the attack, and rank the possible values of the subkey from most to least likely.

The 'partial' refers to the fact that we are finding the guessing entropy on each subkey. This gives us a PGE for each of the 16 subkeys[3]. A PGE of 0 indicates the subkey is perfectly known, a PGE of 10 indicates that 10 guesses were [incorrectly] ranked higher than the correct guess.

The PGE for each subkey is calculated when the attack algorithm has access to $1, 2, \cdots, N$ traces. We record the number of traces when the maximum PGE across all subkeys falls below 10. To improve consistency the PGE for each subkey is averaged over several attacks (trials).

3 Varying Clock Frequency

When an attacker is recording the power traces, ideally each trace would be perfectly synchronized with each

[1] www.chipwhisperer.com
[2] http://avrcryptolib.das-labor.org
[3] This paper is always using AES-128

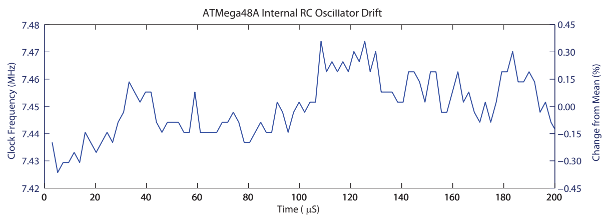


Fig. 2 Atmel AtMega48A internal clock drift during a side-channel attack.

other. That is to say that each time instance across all traces corresponds to the same instruction occurring on the DUT. In real systems, traces may not be perfectly synchronized. This could come from jitter in the trigger signal, unintended non-linear code flow such as interrupts on the DUT, or countermeasures such as instruction shuffling or random delay insertion. A discussion of algorithms and their performance for resynchronizing is compared in [6]. For all these events the clock is operating at a constant frequency.

Another class of synchronization aims to compensate for the clock frequency of the device varying (called *varying clock* or VC), either due to countermeasures or simply due to the oscillator drift. For an example of the natural variation see Fig. 2, which was measured the short-term drift of the internal oscillator on the experimental platform used here. This small amount of variance was enough to prevent the same CPA attack from being successful with over 2500 traces[4], when with a stable crystal oscillator it was successful in only 30 traces. Algorithms which aim to reverse the VC are given in [17, 7, 14, 16].

When a large number of points are required per trace or a large offset from the trigger to the points of interest exist, even the short-term drift differences between the oscillator in the DUT and the oscillator in the oscilloscope may result in desynchronized traces.

With synchronous sampling, variations in clock frequency will naturally be eliminated from the data source. Each sample no longer corresponds to a time instant, but instead to a clock transition. Synchronization may be required for reasons previously discussed such as trigger jitter or countermeasures, but is not needed to compensate for the clock frequency changing.

3.1 Synchronous Sampling of Varying Clock

As a demonstration of synchronous sampling under VC conditions the AtMega48A target was designed to randomly vary the internal clock frequency before calling

[4] After 2500 traces the average PGE was 40, and only 4 of the 16 bytes had a stable PGE < 5

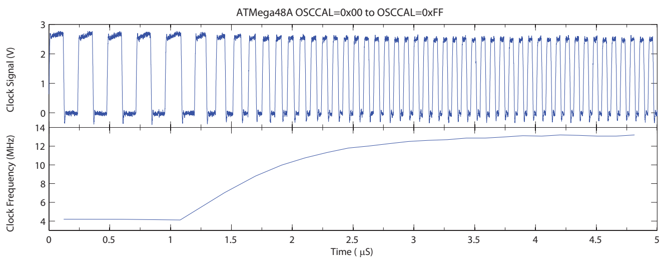


Fig. 3 Atmel AtMega48A internal clock frequency change as OSCCAL changes from 0 to 255.

the AES encryption routines, and a side-channel attack was mounted. For this initial test the CLKOUT fuse was used to output the internal clock onto an IO pin, and the sampling is done synchronous to this clock.

3.1.1 Internal Oscillator Adjustment Range

The AtMega48A datasheet guarantees the oscillator can be calibrated between 7.3 MHz–8.1 MHz, but the actual range is much larger—the specific part used here had a range of 3.95 MHz–13.0 MHz. This test is operating the device outside of guaranteed operating range; commercial products would be advised to only use the adjustment over a smaller range. The time required to switch from the two possible extremes of the randomly selected frequencies, 3.9 MHz to 13 MHz, is shown in Fig. 3. The datasheet specifies a maximum change of 2% clock cycle period between cycles for an external clock; it is not clear if this rapidly changing internal oscillator would also be subject to these considerations[1]. For this reason a number of NOP instructs are inserted before beginning further processing after changing the OSCCAL register.

3.1.2 Internal Oscillator Ranges Used

In this course of this paper, three 'ranges' are used for adjustment of the internal oscillator. The first is the *extended* range, as mentioned spans from 3.9 MHz – 13 MHz. A smaller *narrow* range is also used, which limits the adjustment to a level consistent with the

Table 1 For the ATMega48A, several different clocking options are used. Two of them purposely vary the frequency of the internal oscillator, one uses the internal oscillator without adjustment, and one uses the standard crystal oscillator.

Name	Range(MHz)	Mean(MHz)	Std-Dev
Extended	3.945 – 12.96	7.210	2.190 MHz
Narrow	7.247 – 8.110	7.663	287.5 kHz
Drift	7.315 – 7.413	7.358	11.78 kHz
Crystal	7.373 – 7.373	7.373	5.469 Hz

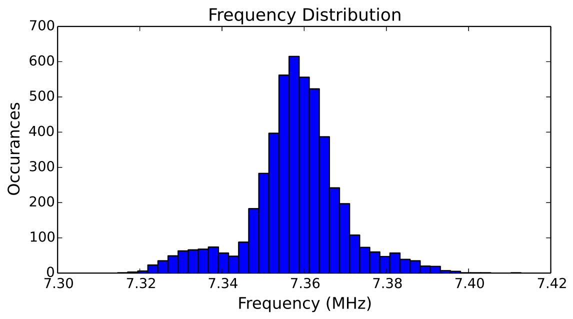


Fig. 4 The histogram of the operating frequency for the 'drift' range. The distribution appears to approximately follow the Normal distribution.

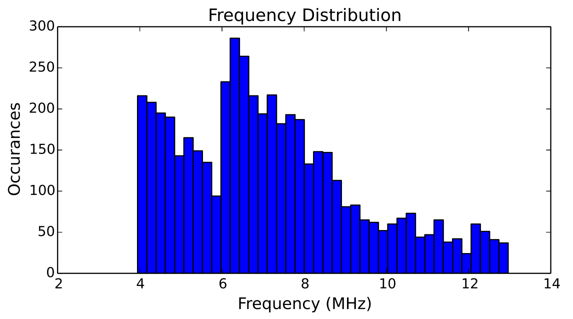


Fig. 5 The histogram of the operating frequency for the 'extended' range. The non-linear mapping of the control register to operating frequency, which is also split into two overlapping ranges, results in a non-standard distribution.

datasheet. Finally the *drift* range is also explored, which reflects the natural random variations due to the nature of the internal oscillator in this device. Detailed information about each of those ranges is presented in Table 1. To validate the frequency measurement system, the *crystal* range is also included, where a crystal oscillator is used to maintain a perfect clock reference.

In Fig. 4, the histogram of operating frequency during the requested encryptions is shown for the 'drift' range. This appears to follow the normal distribution, as would be expected by a process resulting from random noise. In Fig. 5 the histogram is shown for the 'extended' operating range. The value written to the adjustment register (OSCCAL) is uniformly random in the range $[0, 255]$. The AtMega48A splits the OSCCAL register into two over-lapping frequency ranges. Furthermore it does not have linear mapping from the OSCCAL register to operating frequency, resulting in a non-standard distribution[1].

3.2 Preprocessing of Traces

The power consumption of a digital device is dependent on the frequency of operation, and this follows a linear

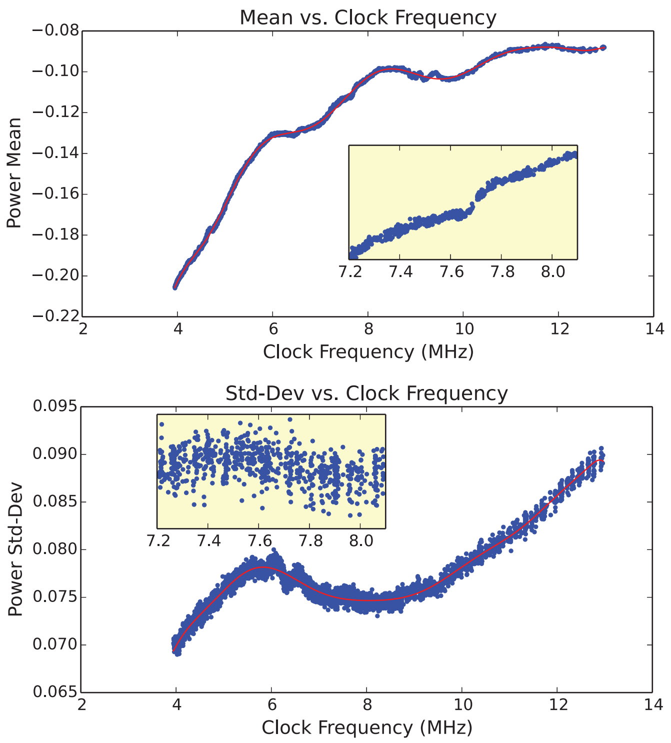


Fig. 6 Plot of the trace mean standard deviation compared to operating frequency of the microcontroller, from 3.9 – 13 MHz. Inset details 7.2 – 8.1 MHz range. Red line shows the $\hat{\mu}(f)$ and $\hat{\sigma}(f)$ functions used in (4) and (5).

relationship. For the ATMega48A at 3.3V, the power consumption when moving from 3.9 MHz to 12 MHz goes from 1.7 mA to 3.1 mA[1]. While the power traces will line up in the time domain with synchronous sampling, they will require scaling in order to allow comparison of the same point across multiple traces. In [14] it is suggested to add an adjustment factor based on the measured frequency of operation, as in (1). Here $T_{p,n}$ is a single point at index p in trace n, C is a scaling constant, and $f_{p,n}$ is the frequency of the clock at point $T_{p,n}$.

$$T'_{p,n} = T_{p,n} + C f_{p,n} \tag{1}$$

This assumes that the change in power measurement due to varying clock frequency simply results in an 'offset' of the measured power. This assumption is also validated in [16], where a 'sliding match' method is used to compensate for the effect of the varying clock on power consumption traces.

To further test this assumption, the mean and standard deviation of each power trace was plotted for the operating frequency f_n, where f_n varies by the 'extended' range given in Table 1. The results are shown in Fig. 6.

Over a somewhat limited range the assumption appears to hold: for example over the range of approximately 7.2 MHz – 8.1 MHz the mean varies linearly with frequency, and the standard deviation is constant. Thus in this range there is no scaling of values, just a bias which must be corrected for. Over the extended frequency range it would appear some scaling of points is required, as the standard deviation is also varying with frequency.

Four additional preprocessing methods are proposed here; all five methods will be tested by comparing the results of the Correlation Power Analysis (CPA) attack over several frequency ranges.

First, two methods which do not require knowledge of the frequency of operation are proposed. The most basic simply scales all traces to be zero-mean, which again would be expected to only work over a limited frequency range:

$$T'_{p,n} = T_{p,n} - \hat{\mu}_{T_n} \tag{2}$$

This can be improved by also scaling by standard-deviation, which should improve performance over a wider range. This will convert the distribution of each trace to be the 'standard normal' distribution. Applying this zero-mean, unit variance normalization (MVN) to side-channel attacks has already been used to improve the applicability of template-based attacks beyond the specific hardware which generated the template[11]. This preprocessing is given by:

$$T'_{p,n} = \frac{T_{p,n} - \hat{\mu}_{T_n}}{\hat{\sigma}_{T_n}} \tag{3}$$

The main downsides of these methods is they require the frequency be constant over the entire length of the trace. Method (1) was proposed in [14] as it could function where the frequency varies per clock cycle. To accomplish this same goal, we will define an estimate function $\hat{\mu}(f)$ which provides an estimate of the mean of the power trace for a known frequency f, and similarly $\hat{\sigma}(f)$ which provides an estimate of standard deviation of the power trace. These functions are simply 15^{th} order polynomial curves fitted to the data in Fig. 6. The plots of both functions are shown in Fig. 6 as well.

Repeating (2) but with $\hat{\mu}$ being a function of f, and not simply calculated over the entire trace:

$$T'_{p,n} = T_{p,n} - \hat{\mu}(f_{p,n}) \tag{4}$$

And similarly for (3):

$$T'_{p,n} = \frac{T_{p,n} - \hat{\mu}(f_{p,n})}{\hat{\sigma}(f_{p,n})} \tag{5}$$

Fig. 7 shows traces before and after preprocessing, using (5)—note the alignment in the time domain of all the traces due to synchronous sampling, despite the varying clock of the DUT.

Even with synchronous sampling, some trace resynchronization may be required. In this case if the sampling was started and then the clock speed changed, the traces had slight misalignment. It is assumed this comes from either the microcontroller delaying execution during the frequency change, or errors in the sampling ADC as the clock frequency changes. The synchronous sampling still greatly simplified the further resynchronization required, as all traces were within 3 samples (clock cycles) of each other. If the sampling was started after the clock frequency speed changed, no resynchronization was required, despite the DUT running at different frequencies.

3.3 Results

The PGE of the CPA attack on an 'extended' frequency variation is shown in Fig. 8. Note from Table 2 the more widely varying 'extended' range of frequency variation has slightly worse performance than the 'drift' range, thus the varying clock does diminish performance. With the crystal oscillator, performance is similar to the 'extended' range. One would expect it to be similar to the 'drift' range instead, since the frequency is not varying. Thus is assumed to be caused by the external oscillator circuitry in the AVR microcontroller resulting in more noise on the trace measurement. Thus using an internal RC oscillator can actually result in lower-noise measurements compared to an external oscillator.

Attempting to attack anything besides the 'crystal' range with measurements taken by a standard asynchronous oscilloscope fails. The PGE does not significantly improve over the range of trace measurements, even for the 'drift' range. This is plotted in Fig 14, where clock recovery is also used.

As previously mentioned a number of trace synchronization methods are also tested, with final results shown in Table 2. The details of the PGE metric are provided in Section 2.1. The $max(PGE) < 10$ point is shown in figures as the horizontal line at $PGE = 10$. It can be noted that over a narrow frequency range no preprocessing is required: the 'drift' range has no improvement using any preprocessing method. Only the

Table 2 The number of traces for the Partial Guessing Entropy (PGE) of the CPA attack to be < 10 is given in this table, where the traces have been preprocessed by different methods.

Clock	$\mathbf{T}_n$	$\mathbf{T}_n + Cf_n$	$\mathbf{T}_n - \mu_{T_n}$	$\frac{\mathbf{T}_n - \mu_{T_n}}{\sigma_{T_n}}$	$\mathbf{T}_n - \hat{\mu}(f_n)$	$\frac{\mathbf{T}_n - \hat{\mu}(f_n)}{\hat{\sigma}(f_n)}$
Extended	93	32	28	30	28	30
Narrow	23	19	16	15	15	15
Drift	12	12	12	13	12	12
Crystal	29	29	29	30	30	29

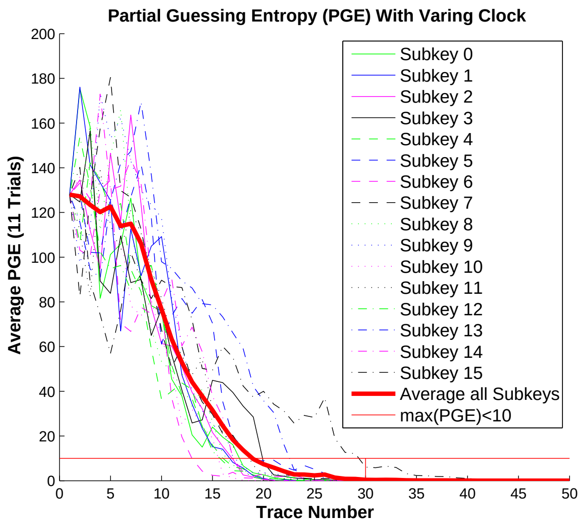


Fig. 8 Results of a CPA attack on a device with oscillator frequency randomly varying between 3.9 MHz–13 MHz on each encryption, and no trace synchronization being performed. The *Byte N* refer to the subkey Partial Guessing Entropy(PGE), *Average* refers to the average of all 16 subkeys. $max(PGE) < 10$ shows the metric used in Table 2.

'extended' range shows significant improvement in attack performance by using preprocessing, and even then the method makes little difference.

These results suggest that details of the preprocessing are not too critical, and would also validate previous work such as [14] which indicate a simple frequency-dependant bias is sufficient. In cases where the frequency is constant over the entire trace, it is sufficient to simply subtract the mean of each trace from itself, forcing the trace to be zero-mean.

Considering the extremely large range the oscillator was varied over (3.9 MHz–13 MHz), these results show that synchronous sampling is a simple method of attacking the varying clock (VC) countermeasure.

4 Clock Recovery

The previous section assumed that the clock was available for the synchronous sampling. In many devices the clock is not available externally, meaning additional work is required to perform this synchronous sampling. In side-channel analysis, it was previously demonstrated how to force an internal oscillator to lock to an external signal [15]. This was used to stabilize the internal RC oscillator and improve trace synchronization, but the same method could be used to generate the reference clock for synchronous sampling. This will fail if the device itself is varying the clock frequency, so instead *clock recovery* must be used to generate a copy of the clock. The idea of clock recovery is not new—in communications electronics this has been used for many years to synchronize a receiver clock to a transmitter clock over long distances[4].

The basic method used here for clock recovery is to filter the power signal so that only the fundamental frequency from the internal oscillator is left. This can then be amplified and turned into a digital signal. To prevent

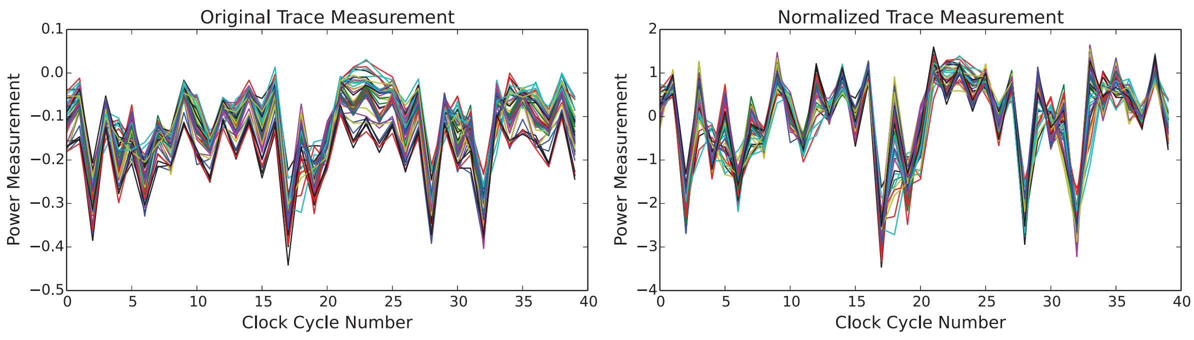


Fig. 7 Traces can be normalized by (5) before passing to a standard CPA attack to remove the effect of varying operating frequency.

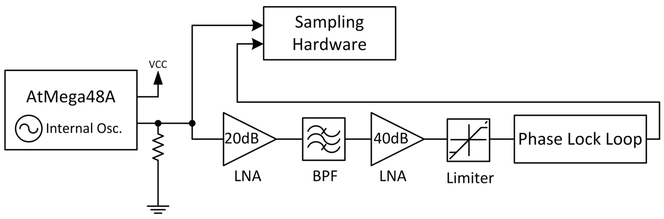


Fig. 9 Clock Recovery Block Diagram.

glitches from resulting at the output a PLL is used to provide a clean digital signal. Details of this hardware design and results of side-channel analysis tests will be presented next.

4.1 Hardware Design

A block diagram of the system is given in Fig. 9, for a complete schematic see Appendix A. A Low Noise Amplifier (LNA) is placed on each side of the band-pass filter (BPF), the BPF selecting the fundamental frequency from the power signal. The output of the final LNA is limited to logic levels and fed into the Phase Lock Loop (PLL) block. The PLL used is a single-chip solution, the Texas Instruments CDCE906 device which integrates the Voltage Controller Oscillator (VCO), Phase Detect (PD), loop filters, and frequency dividers into a single package. For an introduction to PLLs the reader is referred to [2].

Fig. 10 shows an example of recovering an internal oscillator on an Atmel AVR ATMega48A device. With this device it is possible to switch on a 'clock out' pin, which allows measurement of the internal RC oscillator signal. The clock recovery logic works equally well with this pin enabled or not, but enabling the pin allows comparison of the recovered clock to the internal oscillator.

4.2 Filter Design

The design of the band-pass filter (BPF) is critical for the success of the clock recovery, details of the design process are given in Appendix A. Selection of the pass-band is based on the frequency of the internal oscillator for the device under attack. If this frequency is not known it can typically be found by viewing the frequency spectrum of the device during operation.

Careful consideration must be given for the group delay of the filter, which changes over frequency. As an example the 6.5 MHz–8.5 MHz BPF used for the ATMega48A device is shown in Fig. 11. The group delay, which is usually measured in time units or phase degree, has been scaled by the frequency to give us a

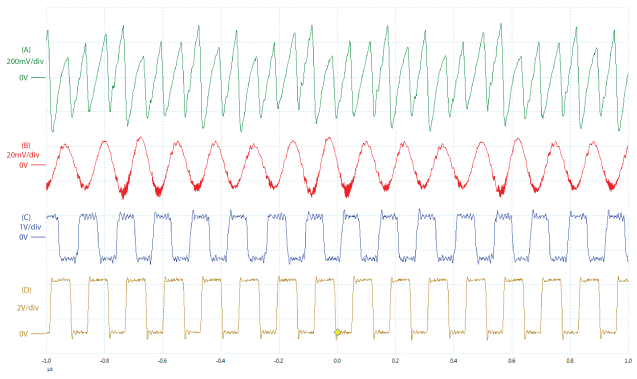


Fig. 10 Recovery of 7.7 MHz Internal RC Oscillator on AT-Mega48A. (A) is the amplified power trace after the LNA. (B) is the output of the band-pass filter, and (C) is the output of the limiter, which generates a logic-level signal. The output of (C) can be passed through a PLL to further stabilize the signal. In (D) the actual RC oscillator output is shown, note the perfect alignment of the recovered signal (C) and internal RC oscillator (D).

group delay in 'clock cycles'. The group delay will cause synchronization errors between traces if the frequency of the DUT oscillator changes, since the delay through the filter varies with frequency.

For more detail, the delay between the actual internal RC oscillator and the recovered clock is plotted in Fig. 12 over a more limited range. Here the delay is measured in degrees, where 360° equals one clock cycle. This figure comes from measurements of the final implemented system, whereas Fig. 11 is based on simulations of just the BPF.

Three methods to reduce this error can be used. First, the type of analog BPF should be matched with the DUT. If the frequency of the oscillator varies only a tiny amount, it would be possible to use a Chebyshev filter with the better attenuation performance. If the DUT oscillator frequency will vary a filter with better group delay performance could be used such as the Bessel. The second way to reduce this error is to measure the frequency during each trace acquisition, and shift the recorded waveform by the known group delay of the filter at this frequency. Finally a standard trace synchronization algorithm can be used to synchronize all such traces.

4.3 Results of CPA Attack

The AtMega48A platform is used again for this evaluation. The 'external clock' output is disabled during these tests—the AVR driving the IO pin at the clock frequency results in a very strong fundamental harmonic on the power trace, which results in a better signal for the PLL to lock onto. Such a system would

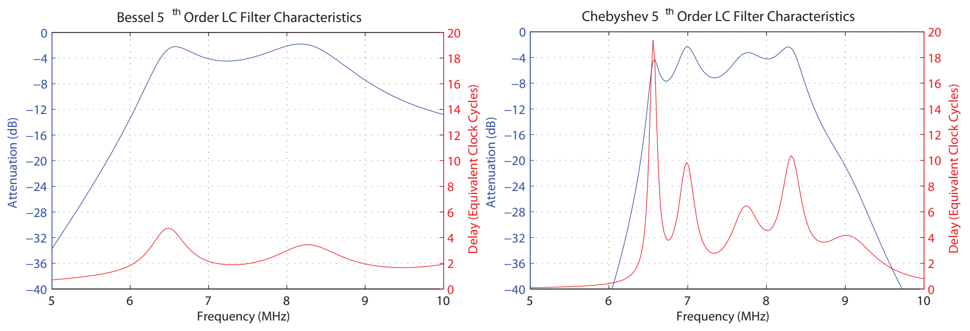


Fig. 11 Choice of filter type means a choice between better group delay performance and better attenuation outside the passband. Two examples are given here: a Chebyshev filter and a Bessel filter, both 5th order made from discrete LC components. Results are from simulation.

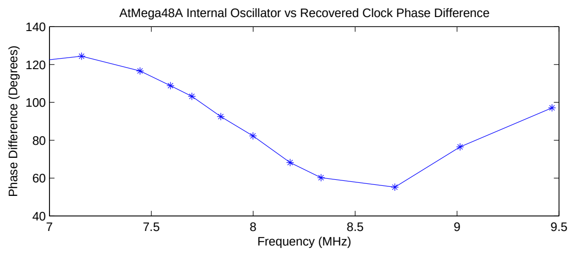


Fig. 12 As the phase difference changes, the alignment of measurements is compromised, requiring more traces. This figure shows the measured phase difference for the overall system, i.e. phase difference between the RC oscillator on the AVR and the final recovered clock. A Bessel analog filter (as given in Appendix A) is used here, results are from measurement.

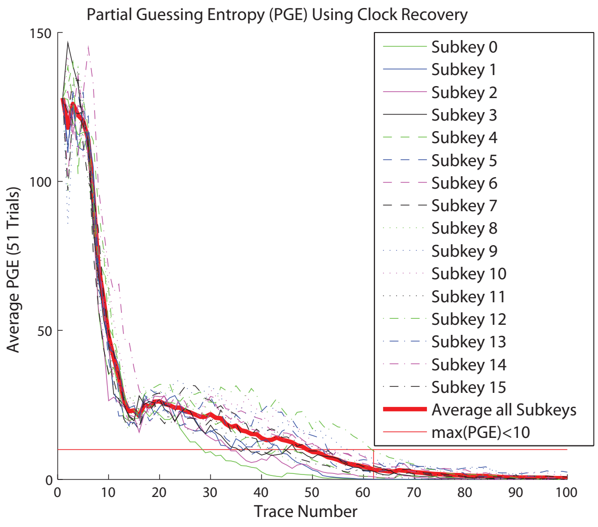


Fig. 14 Results of a CPA attack on a device with an internal RC oscillator, where the oscillator frequency changes ±0.5% during operation due to drift, and the clock is not externally available, but clock recovery with synchronous sampling used. *Average* refers to the average of all 16 subkeys. Subkey plot legend same as in Fig. 8.

be unrealistic since real systems would not be driving an arbitrary IO pin causing this strong fundamental.

The complete setup with clock recovery module, OpenADC capture hardware, and target is shown in Fig. 13.

Initially, the internal RC oscillator is used without any explicit random frequency generation. The RC oscillator does randomly drift about ±0.5% during operation as measured in Table 1. Measurements taken with a standard oscilloscope fail to recover the key as shown in Fig. 16, even after 1000 trace measurements. When the clock is stable the standard oscilloscope recovers the key in <20 traces, as in Fig. 1. Thus the small amount of clock variation causes the CPA attack to fail, despite the starting point having perfect synchronization. If instead we use syncronous sampling with clock recovery as proposed in this paper, the results are as in Fig. 14. The CPA attack is successful without any special processing of the traces.

Next, the 'narrow' frequency range in Table 1 is used for clock recovery, which has a center frequency of 7.66 MHz. The frequency was varied approximately ±5.5%. Fig. 15 gives the results of the CPA attack on this system. The reduced performance is mainly due to the phase delay of the clock varying with frequency, as in Fig. 12. When the clock is directly available and not obtained through clock recovery, as in the results of Table 2, the 'narrow' frequency range has similar performance to the 'drift' range.

The 'extended' clock frequency range of 3.9 MHz – 13 MHz could not be recovered using the simple filtering method. This is due to the fact that the 3^{rd} harmonic of 3.9 MHz will be at 11.7 MHz, which would fall within the bandpass filter bandwidth. Using clock recovery on a very widely varying clock would require a tunable filter which follows the fundamental frequency.

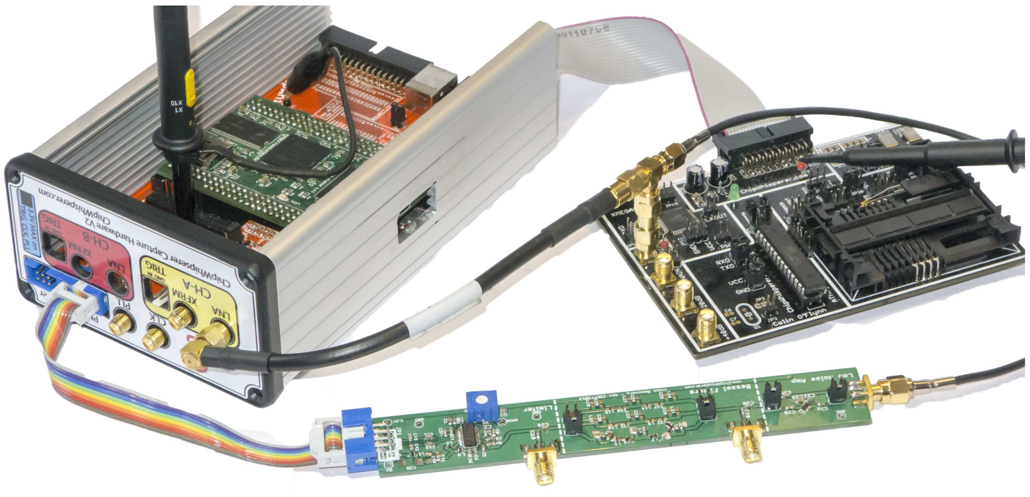


Fig. 13 Test Setup for side-channel analysis with clock recovery of internal oscillator on ATMega48A. The oscilloscope is used to measure recovered clock frequency. The long board center-front performs amplification, filtering, and limiting. The PLL is located inside the capture hardware on the left-hand side. The back right board is the AtMega48A target.

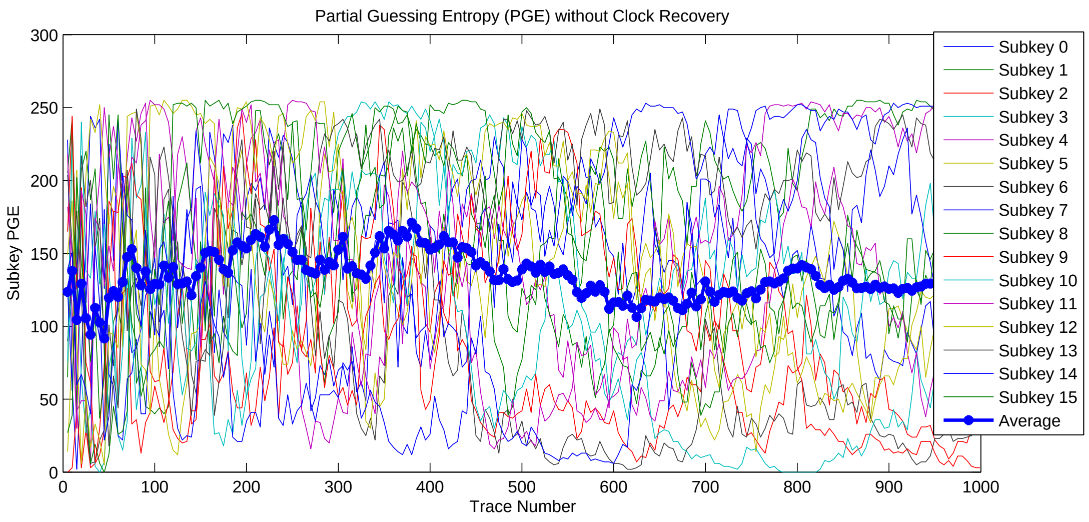


Fig. 16 Results of a CPA attack on a device with an internal RC oscillator, where the oscillator frequency changes ±0.5% during operation due to drift, and a standard *asyncronous* oscilloscope samples the device at 312 MS/s. *Average* refers to the average of all 16 subkeys, and *ASync* refers to the Average PGE for all subkeys of a CPA attack mounted on the same platform but using an oscilloscope sampling at 500 MS/s. Subkey plot legend same as in Fig. 8.

4.4 Software-Only Clock Recovery

Although this work has presented the use of synchronous sampling hardware clock recovery blocks, much of this work can be done in software to allow existing capture systems to benefit from these methods. In this case the 'clock recovery' block diagram in Fig. 9 can be completely implemented in software: the 'band-pass filter' is simply an FIR filter with the input being the measured power traces, and the 'limiter' is a zero-crossing detector. The location of rising edge zero crossings correspond to the samples in the original power trace corresponding to a clock edge.

The software-only solution will simplify the use of correction factors, since issues such as the 'phase delay' of the filter can be perfectly characterized, and more easily compensated for since the original signal was captured at a very high sampling rate. In addition more complex aspects such as filtering on the fundamental frequency, and not some harmonic, may be more easily handled.

The downside of the software implementation is that the original sample rate must be sufficiently high to sample the information at the clock edge. Of course the hardware implementation presented in this paper could be used for both fault injection along with side-

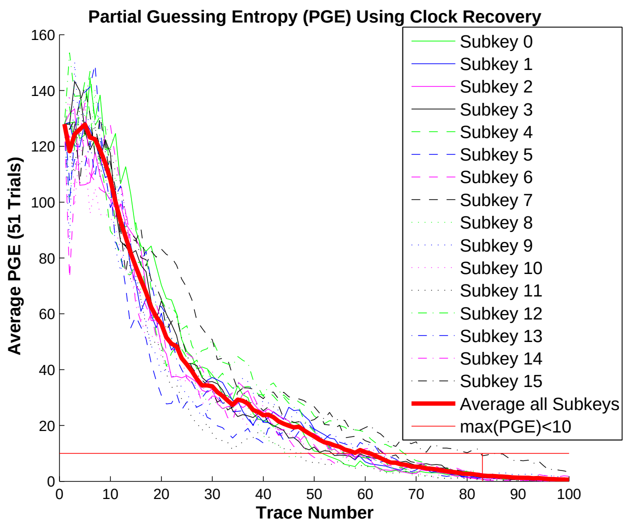


Fig. 15 Results of a CPA attack on a device with an internal RC oscillator, where the oscillator frequency changes ±5.5% during operation, and the clock is not externally available, but clock recovery with synchronous sampling used. Plot legend same as in Fig. 8.

channel analysis, whereas a software-only implementation is only applicable for side-channel analysis.

4.5 Further Work

Preprocessing to account for group delay in the filter may improve results, especially if the clock varies over a wider frequency range. In general this can be accomplished using standard trace synchronization schemes, which may also require sampling at a multiple of the clock frequency for increased time granularity.

Work on side-channel analysis of wireless devices has already been presented [8], using standard AM demodulating techniques. In a similar manner recovery of the clock from this waveform is also possible, and clock recovery may allow attacking more complex devices by providing a clock reference for synchronous sampling. There are many tightly integrated RF SoC devices, such as IEEE 802.15.4 radios, which integrate a microcontroller or AES hardware on the same die as a radio device. Careful design of a proper analog front-end, including the use of clock recovery, could make it possible to detect power variations modulated onto the RF carrier emitted by these devices.

5 Conclusions

Synchronous sampling has already been demonstrated to be a useful tool in reducing the data complexity when working with side-channel analysis measurements [13] [12]. It is know that *compression* of the power traces can be performed post-capture to reduce them to points of interest. Using synchronous sampling, however, eliminates the processing requirement.

Synchronous sampling depends on the availability of the device clock, where many real devices contain an internal oscillator with no external signal. This paper has demonstrated how a 'clock recovery' technique can generate an external reference clock which is phase-locked to the internal oscillator of the device.

If the device under attack is varying the internal oscillator, this external clock will remain phase-locked to the true frequency. As synchronous sampling is measuring clock edges and not absolute time, this varying clock has very little effect of the success rate of an attack performed on these traces. The traces remain perfectly synchronized despite the changing clock frequency.

Beyond improving the correction factor, tests into attack algorithms which do not depend on the absolute peaks could be useful with synchronous sampling. Frequency-Based analysis such as at [5] could be used, where the synchronous sampling would have automatically scaled all frequency components to be relative to the master clock.

Acknowledgements Special thanks to funding provided by NSERC Canada Graduate Scholarship and OZ Optics.

References

1. Atmel Corporation: ATmega48A Datasheet
2. Banerjee, D.: PLL Performance Simulation and Design Handbook, 4th edn. Texas Instruments (2006)
3. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. Cryptographic Hardware and Embedded Systems - CHES 2004 pp. 135–152 (2004)
4. Costas, J.: Synchronous Communications. Communications Systems, IRE Transactions on **5**(1), 99 –105 (1957). DOI 10.1109/TCOM.1957.1097490
5. Gebotys, C.H., Ho, S., Tiu, C.C.: EM Analysis of Rijndael and ECC on a Wireless Java-Based PDA. In: Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings, *Lecture Notes in Computer Science*, vol. 3659, pp. 250–264. Springer (2005). DOI 10.1007/11545262_19
6. Guilley, S., Khalfallah, K., Lomne, V., Danger, J.L.: Formal Framework for the Evaluation of Waveform Resynchronization Algorithms. In: Proceedings of the 5th IFIP WG 11.2 International Conference on Information Security Theory and Practice, WISTP'11, pp. 100–115. Springer-Verlag, Berlin, Heidelberg (2011). URL `http://dl.acm.org/citation.cfm?id=2017824.2017835`
7. Kafi, M., Guilley, S., Marcello, S., Naccache, D.: Deconvolving Protected Signals. In: Availability, Reliability and Security, 2009. ARES '09. International Conference on, pp. 687 –694 (2009). DOI 10.1109/ARES.2009.197

8. Kasper, T., Oswald, D., Paar, C.: Side-channel Analysis of Cryptographic RFIDs with Analog Demodulation. In: Proceedings of the 7th international conference on RFID Security and Privacy, RFIDSec'11, pp. 61–77. Springer-Verlag, Berlin, Heidelberg (2012). DOI 10.1007/978-3-642-25286-0_5. URL `http://dx.doi.org/10.1007/978-3-642-25286-0_5`
9. Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Advances in Cryptology - CRYPTO' 99, pp. 388–397. Springer-Verlag (1999)
10. Massey, J.: Guessing and entropy. In: Information Theory, 1994. Proceedings., 1994 IEEE International Symposium on, pp. 204– (1994). DOI 10.1109/ISIT.1994.394764
11. Montminy, D., Baldwin, R., Temple, M., Laspe, E.: Improving cross-device attacks using zero-mean unit-variance normalization. Journal of Cryptographic Engineering **3**(2), 99–110 (2013). DOI 10.1007/s13389-012-0038-y
12. O'Flynn, C., Chen, Z.D.: A case study of Side-Channel Analysis using Decoupling Capacitor Power Measurement with the OpenADC. Lecture Notes in Computer Science **7743**, 328–344 (2013)
13. O'Flynn, C., Chen, Z.D.: Chipwhisperer: An open-source platform for hardware embedded security research. In: Constructive Side-Channel Analysis and Secure Design - COSADE 2014 (2014)
14. Réal, D., Canovas, C., Clédière, J., Drissi, M., Valette, F.: Defeating Classical Hardware Countermeasures: A New Processing for Side Channel Analysis. In: Proceedings of the Conference on Design, Automation and Test in Europe, DATE '08, pp. 1274–1279. ACM, New York, NY, USA (2008). DOI 10.1145/1403375.1403684
15. Skorobogatov, S.: Synchronization method for SCA and fault attacks. Journal of Cryptographic Engineering **1**(1), 71–77 (2011). DOI 10.1007/s13389-011-0004-0
16. Tian, Q., Huss, S.: On Clock Frequency Effects in Side Channel Attacks of Symmetric Block Ciphers. In: New Technologies, Mobility and Security (NTMS), 2012 5th International Conference on, pp. 1 –5 (2012). DOI 10.1109/NTMS.2012.6208680
17. van Woudenberg, J.G.J., Witteman, M.F., Bakker, B.: Improving Differential Power Analysis by Elastic Alignment. In: Proceedings of the 11th International Conference on Topics in Cryptology: CT-RSA 2011, CT-RSA'11, pp. 104–119. Springer-Verlag, Berlin, Heidelberg (2011)
18. Yang, S., Gupta, P., Wolf, M., Serpanos, D., Narayanan, V., Xie, Y.: Power analysis attack resistance engineering by dynamic voltage and frequency scaling. ACM Trans. Embed. Comput. Syst. **11**(3), 62:1–62:16 (2012). DOI 10.1145/2345770.2345774

Appendix A: Hardware and Design Details

This appendix provides some brief notes on the physical hardware realized in this paper, along with a few notes for researchers looking to duplicate it. Note that full details are posted as part of the ChipWhisperer Wiki at `http://www.ChipWhisperer.com`.

5.1 Core Clock Recovery Module

The core part of this work is a module with a Low Noise Amplifier (LNA), Limiter, and Phase-Lock Loop (PLL) chip. The schematic for this is given in Fig. 17. The LNA is an Analog Devices AD8331, which has a variable gain up to 55dB. A resistor connected to the 'RLIM' pins provides an ability to set an arbitrary clipping level for the output. This clipped output is connected to the PLL chip, which is a Texas Instruments CDCE906. The clipped output from the LNA is used a LVDS input to the PLL, which works assuming the input to the entire block was sufficiently clean, that is to say contains only a single frequency component. Additional filtering can be added by placing capacitors on each of the input pins of the CDCE906 to ground, values between 100 pF–680 pF are reasonable depending on the fundamental frequency being targeted.

The CDCE906 was chosen for it's ability to operate down to 1 MHz, many PLL devices have higher lower frequency limits. If attacking devices with relatively slow internal oscillators, such as the KeeLoq devices at 1.3 MHz, this lower range is needed. The CDCE906 can be configured via I^{2}C to adjust parameters such as input drive level, frequency divider settings, and outputs in use. For this work it was configured to enable the PLL with frequency dividers such that the input and output frequency were the same. The sampling rate can easily be set to a higher multiple of the system frequency with this PLL block.

5.2 Filter

The filter design was done using the Quite Universal Circuit Simulator (QUCS) software. QUCS contains a *Filter Synthesis* tool, which can be used to generate an appropriate band-pass filter. This will be calculated with 'ideal' component values, and then these values are adjusted to the closest standard part, and a simulation confirms if the performance is still acceptable.

Note that at DC the filter will present a dead short, as no blocking capacitors are present. If connecting one side of the filter to a shunt or other device with a DC bias, always insert DC blocking capacitors.

5.3 First Stage LNA

An additional LNA may be required in front of the band-pass filter depending on the signal strength. It is possible to use a standard device such as a MiniCircuits *ZFL-1000LN+*. Care must be taken with RF amplifiers, as most of them are designed for use with 50Ω

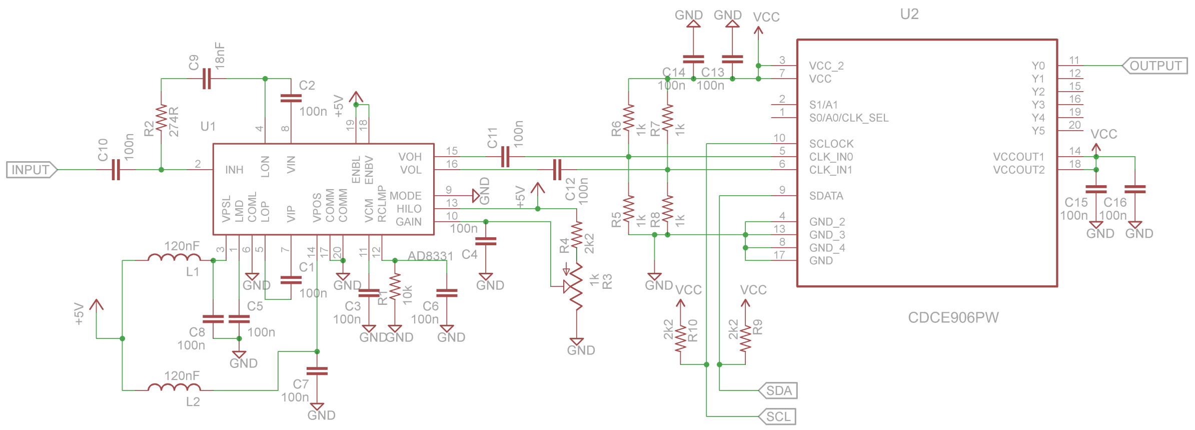
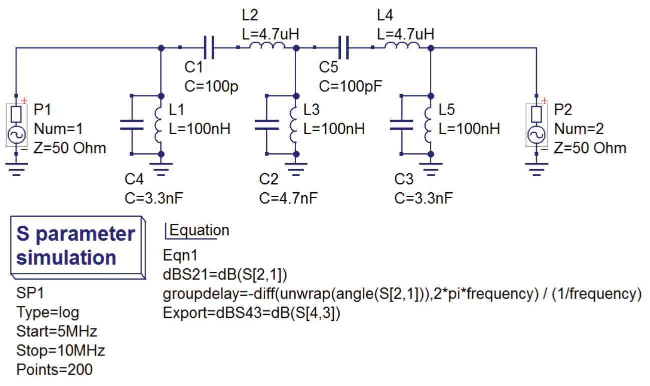


Fig. 17 Schematic for the LNA, Limiter, and PLL as used in Fig. 9.



SP1
Type=log
Start=5MHz
Stop=10MHz
Points=200

Eqn1
dBS21=dB(S[2,1])
groupdelay=-diff(unwrap(angle(S[2,1])),2*pi*frequency) / (1/frequency)
Export=dBS43=dB(S[4,3])

Fig. 18 Bandpass Filter Design Environment. Note the component values have been changed to reflect those being used in the actual circuit, and some optimizations may be needed to get acceptable performance. The equation to plot group delay in clock cycles can be seen in this diagram.

systems. If the output or input is not matched properly the amplifier may oscillate, causing errors. Generally amplifiers based on Op-Amps are safer in this regard, and specially-designed differential amplifiers can be exceedingly useful when measuring across current shunts.