Chapter 9

Experiment: Use the KRYPTON CPLD card to capture and display analog signals

In this experiment, we will use the Krypto CPLD to build a mixed-signal system which can sample and display an analog signal. The overall plan for the system is shown in Figure 9.1. Briefly,

- The analog input is converted to digital form using an analog-to-digital converter (ADC) using a sampling rate of 1KHz. Up to 8192 samples of the input signal will be collected.
- The digital subsystem will collect 8192 samples of the ADC input and store the samples in SRAM in wrap around fashion. The digital subsystem will control the sampling rate etc. for the ADC by using the ADC interface signals (this protocol will be described in detail in Section 9.1). When instructed to do so, the digital subsystem will read out the sampled values from the memory in wrap around read fashion and send them to the digital-to-analog converter (DAC) at a rate of one sample every $1ms$ (i.e. at 1KHz).
- The DAC will convert the input digital code-word to an analog signal which can be displayed using a digital storage oscilloscope.

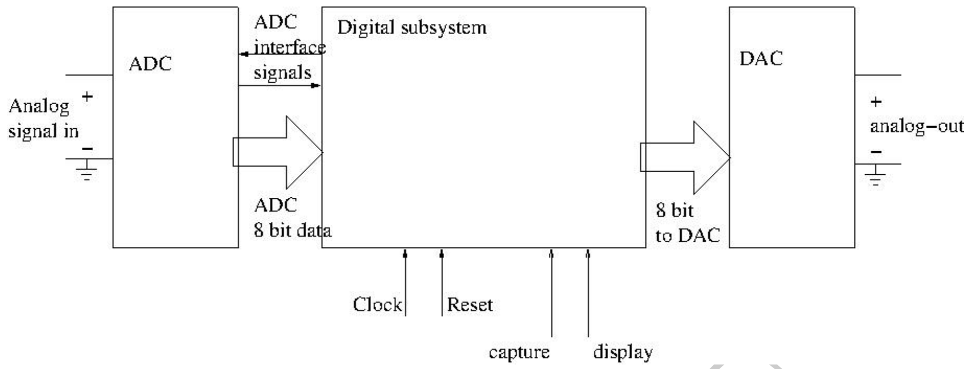


Figure 9.1: System to be implemented

9.1 Using the ADC0804 Analog-to-digital converter

The ADC080x family is a CMOS 8-bit Successive Approximation Analog to Digital Converter (ADC). It can be easily interfaced to any micro-controller, FPGA or CPLD board as the logic levels meet both TTL and MOS specifications. It can convert an analog signal in the range of 0-5 V to a valid digital output. It is packaged in a 20-pin DIP. The pin diagram of ADC 0804 is shown in Figure 9.2

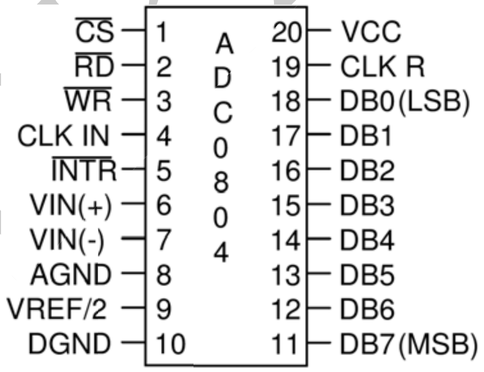


Figure 9.2: Pin diagram of ADC0804

9.1.1 Hardware Interfacing of the ADC0804 with the Krypton CPLD card

The ADC0804 can be interfaced with Krypton CPLD card as shown in Figure 9.3. The pins $\overline{CS}$, $\overline{RD}$, $\overline{WR}$ and $\overline{INTR}$ are the interface control pins for the ADC, and D7 downt to D0 are the digital outputs. The $\overline{CS}$ pin is used to enable the IC, $\overline{WR}$ and $\overline{RD}$ are used to select the operation of sampling the input analog value reading the digital value from the output respectively. The $\overline{INTR}$ is a signal from the ADC to the controller to indicate the completion of conversion. The digital output from the ADC can be displayed using the LED's on the Krypton CPLD card.

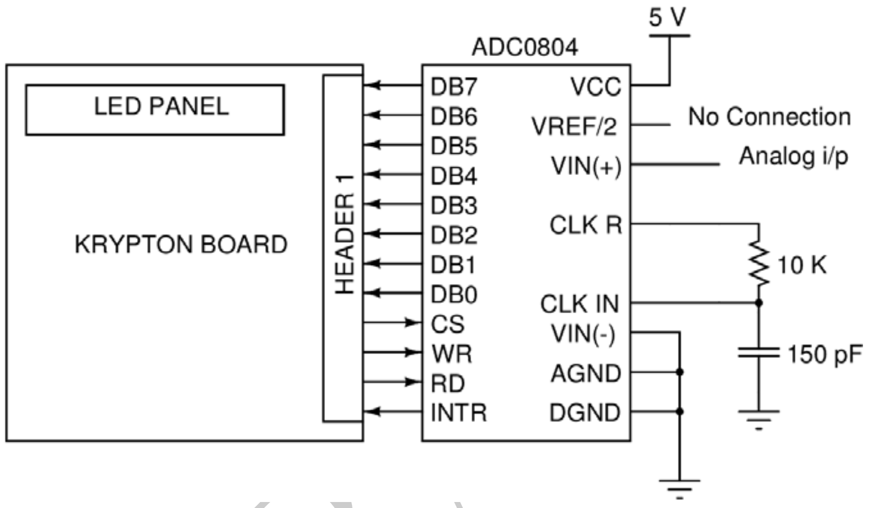


Figure 9.3: Interfacing ADC0804 to Krypton

The voltage at Vref/2 (pin 9) of the ADC0804 can be externally adjusted to convert smaller input voltage ranges to full 8 bit resolution. If Vref/2 (pin 9) is left open, the input voltage range is 0 - 5 V and the step size is $\frac{5V}{255}$=19.6 mV. In Table 9.7, we show the different Vref/2 voltages and corresponding analog input voltage ranges. In this particular exercise we leave pin 9 open to get an input voltage range of $0V$ to $5V$.

Note: Do not use on board supply for $5V$. Use power supply for $5V$.

Table 9.1: Voltage given at Vref/2 pin and corresponding input voltage ranges

Vref/2 (V)	Input Voltage Range (V)	Step Size (mV)
Left Open	0-5	19.6
2	0-4	15.69
1.5	0-3	11.76
1.28	0-2.56	10.04
1	0-2	7.84
0.5	0-1	3.92

9.1.2 The interface timing diagram for the ADC0804

The timing diagram of the ADC0804 is shown in Figure 9.4.

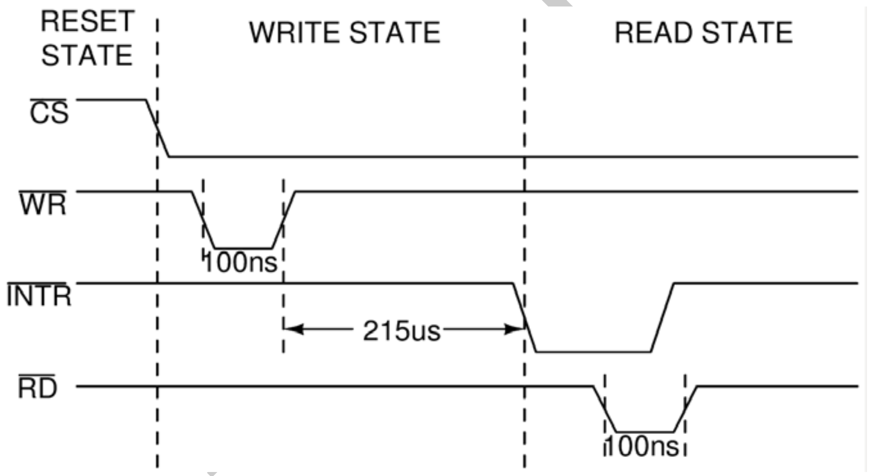


Figure 9.4: Timing diagram of ADC0804. The timings indicate the minimum durations between events.

The IC has a active low *chip-select* ($\overline{CS}$) pin. When reset is asserted, $\overline{CS}$, $\overline{RD}$ and $\overline{WR}$ are made high. To sample the analog signal for conversion, a pulse which goes from high to low is applied on the $\overline{WR}$ pin. The conversion starts after the application of the pulse. The completion of conversion is indicated by $\overline{INTR}$ pin going low. Once the $\overline{INTR}$ goes low we apply a pulse going from high to low on the $\overline{RD}$ pin. This completes one read-write cycle. The same cycle of sampling the input data, converting it and reading at the output is repeated over and over again at the required sampling frequency (1 KHz in our case).

Table 9.2: Pin mapping of HEADER1 with CPLD

Even No. Pin	Odd No. Pin
(Header Pin 40) VDD (5V)	GND
108	105
107	103
106	101
104	97
102	95
98	93
96	89
94	87
88	85
86	81
84	79
80	77
76	75
74	73
72	71
70	69
68	67
66	63
62	59 (Header Pin 1)

9.2 Using the Hitachi HY62256A SRAM

The HY62256A SRAM has 15 address lines and 8 bidirectional data lines. Address lines from 0 to 12 are used for the experiment and hence we can access 8192 address locations from SRAM. Data lines are used for writing data in write mode as well as reading data in read mode. The supply voltage range for the SRAM is 2V-5V. It is packaged in 28 pin DIP. The pin diagram is shown in Figure 9.5.

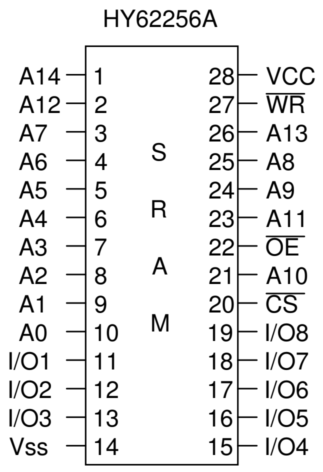


Figure 9.5: Pin diagram of SRAM HY62256A

9.2.1 Hardware Interfacing of the SRAM with the Krypton CPLD card

Hardware interfacing of SRAM with krypton board is as shown in 9.6. Pins A0 through A12 are used to access 8192 address locations in read or write mode. Pins I/O1 through I/O8 are bidirectional pins which can be used to write/read data to/from SRAM. The mode of operation is defined by active-low control signals $\overline{CS}$, $\overline{WE}$ and $\overline{OE}$ as shown in Table 9.3.

Table 9.3: Mode of operation of SRAM

$\overline{CS}$	$\overline{WE}$	$\overline{OE}$	Mode	I/O operation
H	X	X	Standby	High-Z
L	H	H	Output Disabled	High-Z
L	H	L	Read	Data Out
L	L	X	Write	Data In

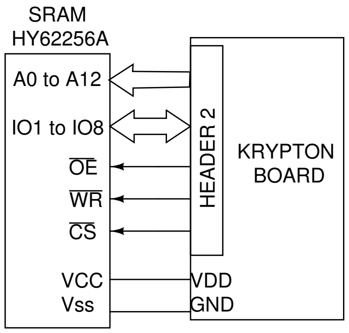


Figure 9.6: Interfacing SRAM to Krypton

9.2.2 Read timing diagram for the SRAM

The read timing diagrams are shown in Figure 9.7, Figure 9.8, and Figure 9.9. The signal $\overline{WE}$ is to be kept high for the read cycle. The ranges of values allowed for the timing parameters are shown in Table 9.4.

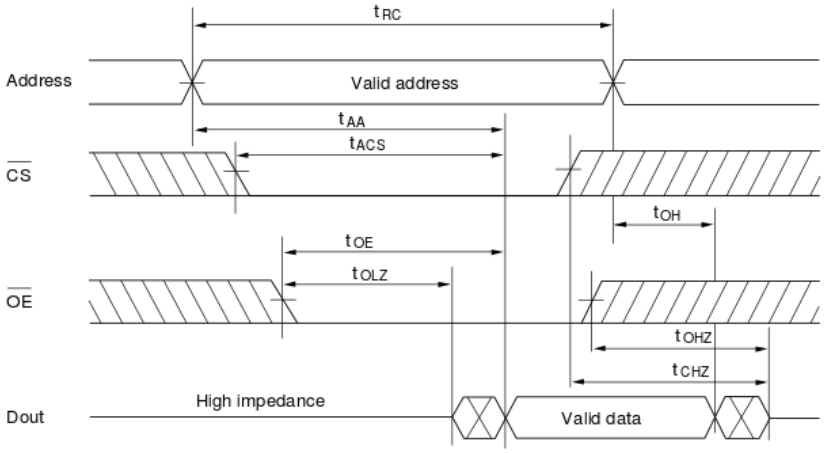


Figure 9.7: Timing diagram of SRAM (Read Mode)

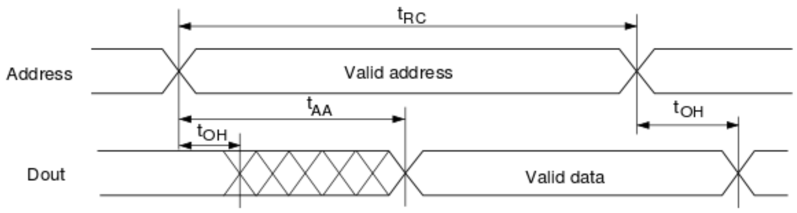


Figure 9.8: Timing diagram of SRAM (Read Mode)

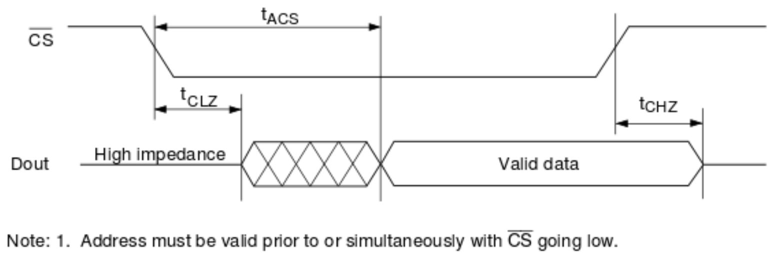

$\overline{CS}$
t_{ACS}
t_{CLZ}
t_{CHZ}
Dout
High impedance
Valid data
Note: 1. Address must be valid prior to or simultaneously with $\overline{CS}$ going low.

Figure 9.9: Timing diagram of SRAM (Read Mode)

Table 9.4: Timing parameters for the Read cycle

Symbol	Parameter	Min (ns)	Max (ns)
t$_{RC}$	Read Cycle Time	45	–
t$_{AA}$	Address Access Time	–	45
t$_{ACS}$	Chip Select Access Time	–	45
t$_{OE}$	Output enable to output valid	–	30
t$_{CLZ}$	Chip Selection to output in low Z	5	–
t$_{OLZ}$	Output enable to output in low Z	5	–
t$_{CHZ}$	Chip deselection into output in high Z	0	20
t$_{OHZ}$	Output disable to output in high Z	0	20
t$_{OH}$	Output hold from address change	5	–

9.2.3 Write timing diagram for the SRAM

Writing is done during the period when $\overline{CS}$ and $\overline{WE}$ are both asserted (made low). The write operation begins at the latest of the low going transitions at $\overline{CS}$ and $\overline{WE}$, and ends at earliest high-going transition at $\overline{CS}$ and $\overline{WE}$. If $\overline{CS}$ goes low simultaneously with $\overline{WE}$ going low, or after $\overline{WE}$ going low, the outputs remain in high impedance state.

The timing diagram for the write is shown in Figure 9.10 and Figure 9.11 respectively. Specified values for the timing parameters are shown in Table 9.5.

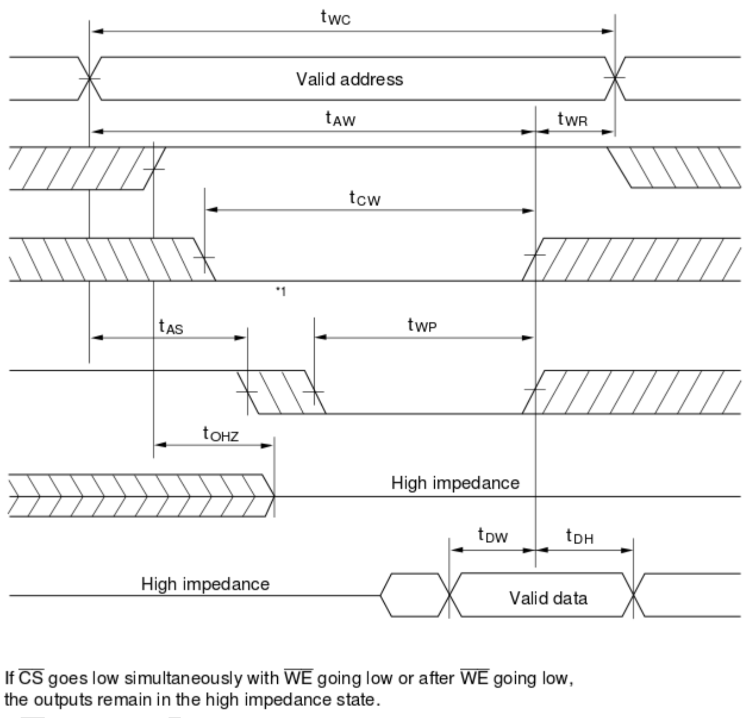
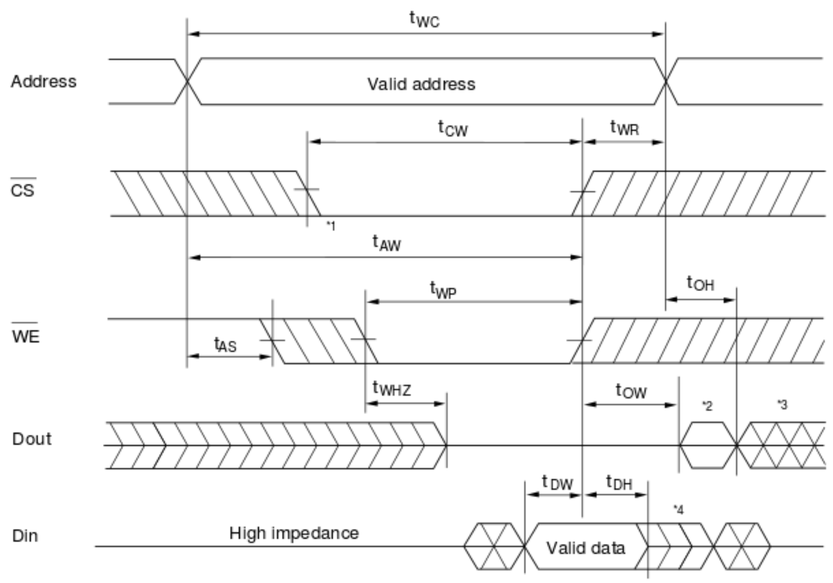


Figure 9.10: Timing diagram of SRAM (Write Mode)



Notes: 1. If $\overline{\text{CS}}$ goes low simultaneously with $\overline{\text{WE}}$ going low or after $\overline{\text{WE}}$ going low, the outputs remain in the high impedance state.

2. Dout is the same phase of the write data of this write cycle.
3. Dout is the read data of next address.
4. If $\overline{\text{CS}}$ is low during this period, I/O pins are in the output state. Therefore, the input signals of theopposite phase to the output must not be applied to them.

Figure 9.11: Timing diagram of SRAM (Write Mode)

Table 9.5: Timing for the Write cycle

Symbol	Parameter	Min (ns)	Max (ns)
t_{WC}	Write Cycle Time	45	–
t_{CW}	Chip Selection to end of write	35	–
t_{AS}	Address Set up Time	0	–
t_{AW}	Address valid to end of write	35	–
t_{WP}	Write pulse width	30	–
t_{WR}	Write Recovery Time	0	–
t_{WHZ}	$\overline{WE}$ to output in high Z	0	20
t_{DW}	data to write time overlap	20	–
t_{DH}	Data hold from write time	0	–
t_{OW}	Output active from end of write	5	–
t_{OHZ}	Output disable to output in high Z	0	20

Note: The SRAM takes 120 ns to complete a write operation. Hence we need to wait atleast 7 clock cycles if we are using 50 MHz clock

Table 9.6: Configuration of PINs to HEADER 2

SRAM_PIN	CPLD_PIN	SRAM_PIN	CPLD_PIN
A14		VCC	VCC(5V)
A12	123	/WE	133
A7	118	A13	
A6	117	A8	119
A5	114	A9	120
A4	113	A11	122
A3	112	/OE	138
A2	111	A10	121
A1	110	/CS	139
A0	109	I/O8	131
I/O1	124	I/O7	134
I/O2	125	I/O6	129
I/O3	130	I/O5	132
GND	GND(0V)	I/O4	127

9.3 Using the digital-to-analog converter: the DAC0808

The DAC0808 is an 8-bit monolithic Digital to Analog Converter (DAC) featuring a full scale output current settling time of 150 ns. It converts 8-bit parallel digital inputs to analog current variations and can be interfaced directly with CMOS logic levels. The pin diagram of the 16-pin DIP DAC0808 package is shown in Figure 9.12.

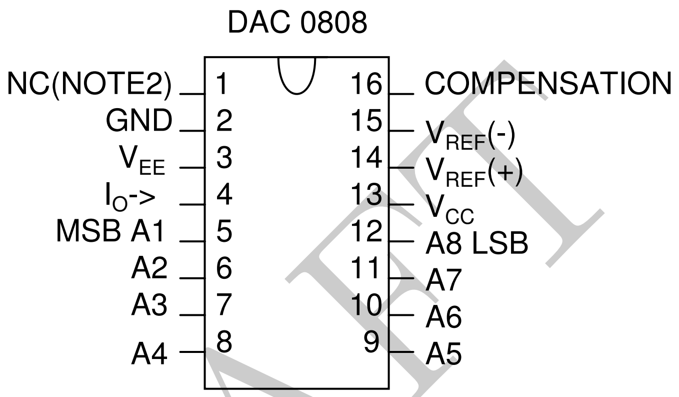


Figure 9.12: Pin diagram of the DAC0808

9.3.1 Hardware Interfacing of the DAC0808 to the Krypton CPLD card

Since the DAC0808 is a parallel DAC, only the 8-bit digital data needs to be connected directly from the Krypton. As shown in the Figure 9.13, 8 pins from HEADER 2 can be connected to the 8 input pins of DAC0808 (pins 5-12).

The output analog variations are in the form of current and it has to be converted in to voltage variations. The LF351, a wide bandwidth operational amplifier is being used for this purpose. The op-amp in inverting configuration will convert the DAC0808 output current to the final voltage variations.

The reference pin V_{REF} can be adjusted to change the output level. The output voltage can be represented as

$$V_{OUT} = V_{REF} \left(\frac{A1}{2} + \frac{A2}{4} + ... + \frac{A8}{256}\right) \tag{9.1}$$

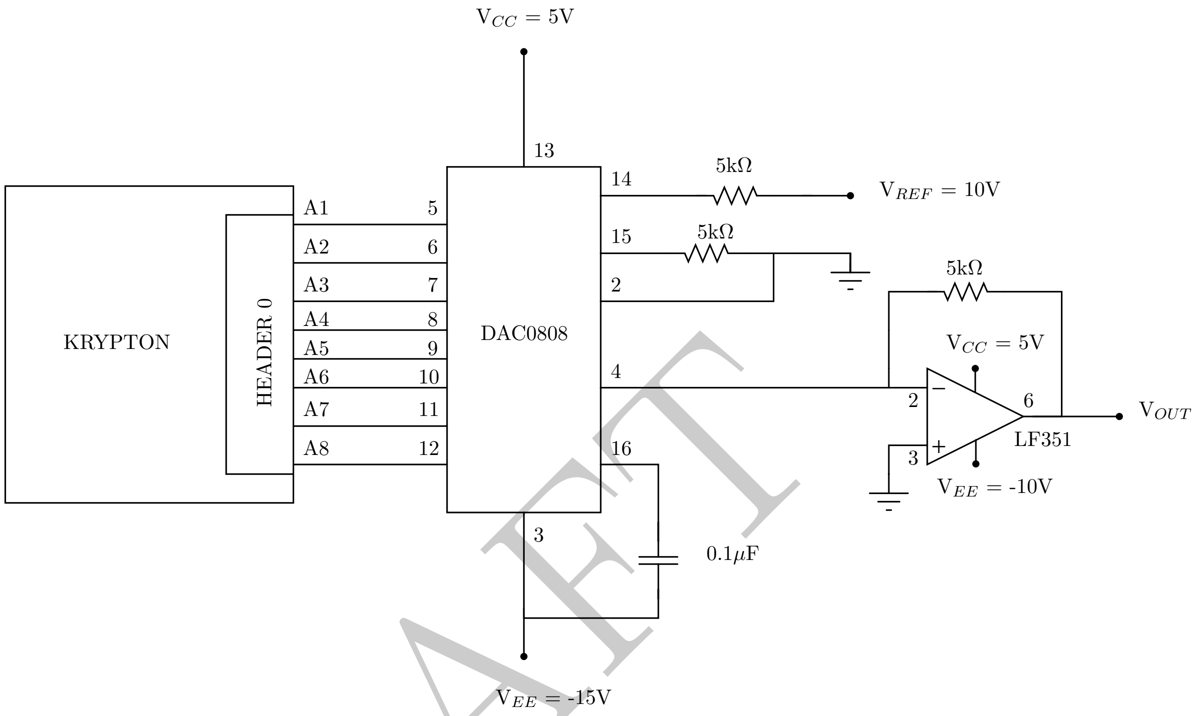


Figure 9.13: Circuit diagram

Table 9.7: Pin mapping of HEADER0 with CPLD

Odd No. Pin	Even No. Pin
(Header Pin 1) 1	2
3	4
5	6
7	12
13	14
15	16
21	22
23	24
27	28
29	30
31	32
37	38
GND	VDD (5V) (Header Pin 26)

9.4 The experiment: a strategy for implementation

To implement the digital component of the overall system shown in Figure 9.1, we recommend that you divide the system into the structure shown in Figure 9.14.

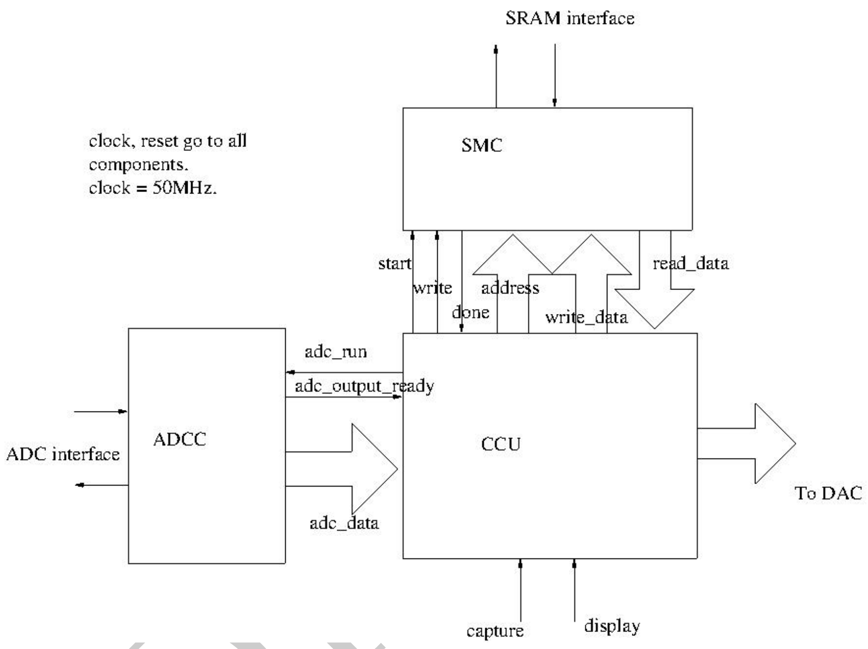


Figure 9.14: Suggested architecture of the digital system

The components of the architecture are:

- The ADC-controller (ADCC) : When *adc_run* is asserted, the ADC-controller starts interacting with the ADC to produce a converted value every $1ms$. The ADC-controller indicates the availability of a converted value using the *adc_output_ready* signal.
  - Note that when the ADCC drives a signal to which the ADC is edge-sensitive, you will need to clean up the signal by registering it (so that glitches in the signal are removed).
- The SRAM-memory-controller (SMC): Has inputs *mc_start*, *mc_write*, *mc_address*, *mc_write_data*, and outputs *mc_done* and mc_read_data (in addition to clock and reset, of course). When *mc_start* is asserted:

  - If at the instant that *mc_start* is sensed true, if *mc_write* is true, the memory-controller initiates a write to the SRAM using the SRAM access signals. When the write is completed, it asserts *mc_done*.
  - If at the instant that *mc_start* is sensed true, if *mc_write* is false, the memory-controller initiates a read from the SRAM using the SRAM access signals. When the read is completed, it asserts *mc_done* and at the same time makes the read data available on *mc_read_data*.
  - Note that when the SMC drives a signal to which the SRAM is edge-sensitive, you will need to clean up the signal by registering it (so that glitches in the signal are removed).

- The central control unit (CCU): This connects to the ADCC, the SMC, and to the DAC pins. In addition, it has two control input called *capture*, *display*. While *capture* is asserted, the CCU asks the ADCC to begin providing samples (one sample every $1ms$), and saves these samples into memory through the SMC (in a wrap-around manner). If *capture* is de-asserted and *display* is asserted, the CCU uses the SMC to access samples from the memory and supplies them to the DAC at a rate of $1Khz$. As long as $display.\overline{capture} = 1$, memory will be read continuously in a wrap-around fashion and the stored values supplied to the DAC at a rate of $1KHz$.

Thus, the overall system can be implemented by

- Specify each subsystem completely: that is, the interfaces, and the internal behaviour.
- Implement each subsystem completely: that is, design the algorithm, implement in VHDL, and simulate each one individually (using dedicated test-benches).
- Perform post synthesis (gate level) simulation for each subsystem.
- Assemble the subsystems into a single system, simulate it and perform post synthesis simulation.
- Assemble the final system: the digital system in the CPLD interfaced to the ADC, DAC and SRAM. Ideally you should simulate this as well, but since you are probably running out of time, you can wire it up, program the CPLD and see if it works.

– Use switches on the CPLD card to implement the *capture*, *display* and *reset* inputs to the digital system.

– Bring some useful signals to the CPLD car LED's so that you can monitor what is happening in the system.