PicoBlaze using Xilinx ISE: Implemented on the Digilent Nexys Board

Fletcher Fleming

Florida State University

Department of Electrical and Computer Engineering

EEL 5707 ASIC Final Project Presentation

Presented on December 1st, 2010, Tallahassee, Florida

Outline

- Project Outline:
  - Description
  - Experimental setup
  - Approach: Literary review & Gantt chart
- Research & Development:
  - Configure development environment
  - Testing the board's peripherals
  - Embedding PicoBlaze
  - "C" to assembly conversion
  - Sum of products (SoP)
  - Analog to digital assessment
- Limitations and suggested future developments
- Intended research extrapolations
- Conclusion

Project Description

Proposal Statement:

"Develop a PicoBlaze application e.g. sum-of-products in a "C" or assembly program:

$$\sum_{k=0}^{k=15} x(k) * f(k)$$

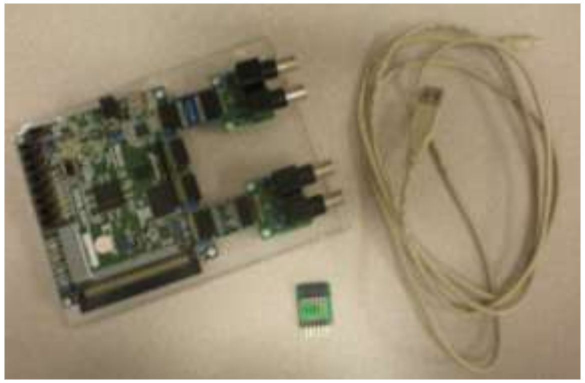
for the design and measure the speed."

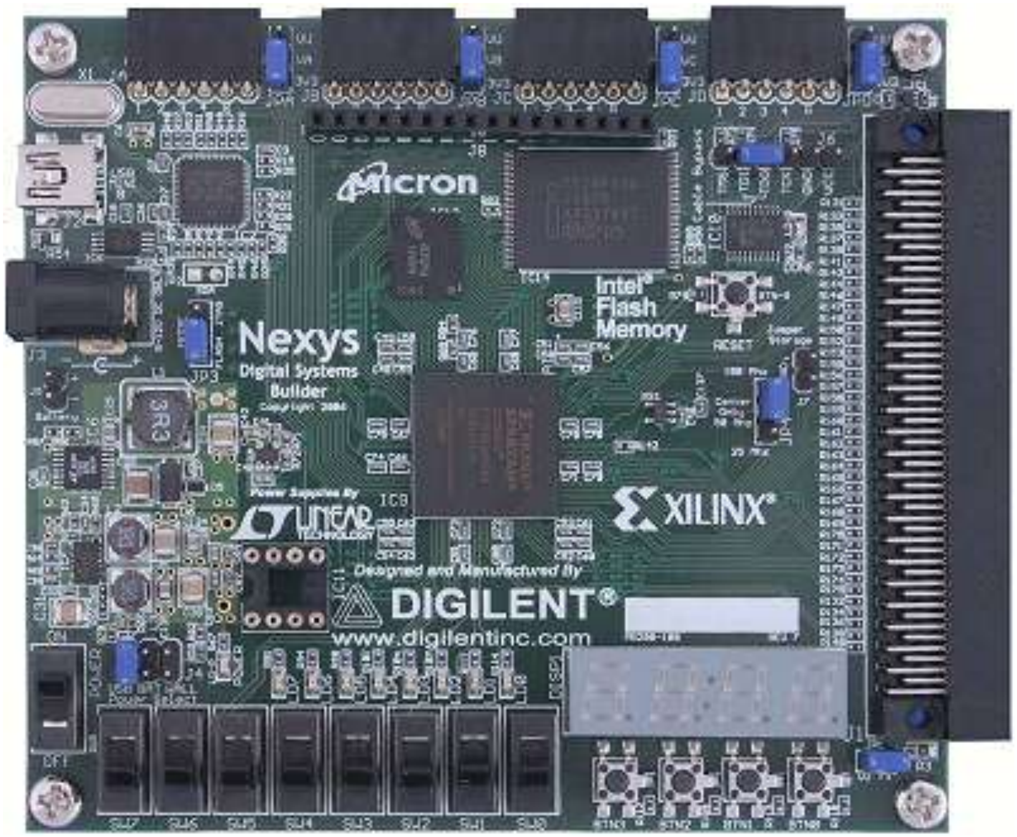
Personal Goals:

Gain Xilinx experience

Play with ADC

Experimental Setup – Hardware: Provided Equipment





Given:

- Digilent Nexys Board
  - discontinued for Nexys2
- Xilinx XC3S200 FPGA
  - FT256 family
- 4 ADC ports connected via 4 6 pin-headers
- USB to Mini-USB cable
- Extra M-F 6 pin connector

Experimental Setup – Hardware: Nexys Specifications

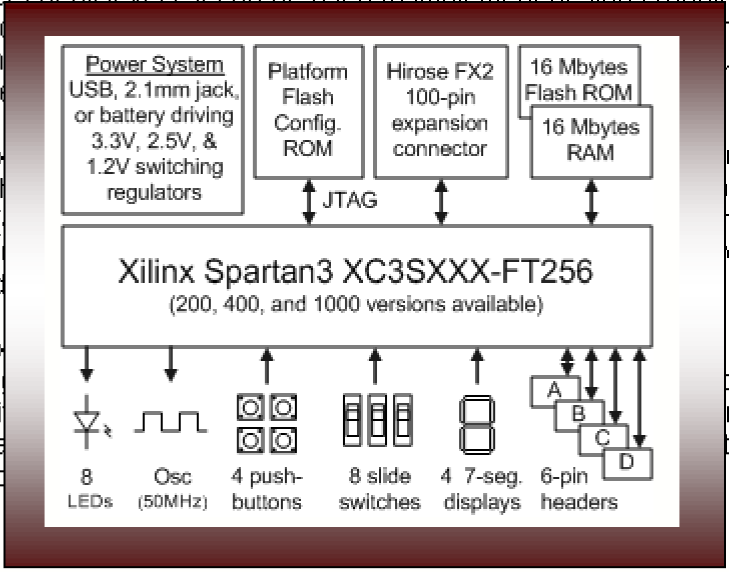
IC:	Xilinx Spartan-3 FPGA (400K gates)
Connector(s):	USB2 Port Hirose FX2 Four 6-pin Pmod connectors PS/2 keyboard/mouse connector
Programming:	JTAG programming via on-board USB2 port (using the free Adept Suite Software); JTAG with JTAG3 parallel cable (JTAG cable not included); numerous FPGA configuration options

- Xilinx XC3S400 FPGA with 500+MHz operation (400K gates)
- USB2 port for FPGA configuration and data transfers (software available as free downloads)
- USB-powered (batteries and/or wall plug can also be used)
- 16MB of fast Micron PSDRAM and 4MB of Intel StrataFlash Flash ROM
- Xilinx Platform Flash ROM that stores FPGA configurations indefinitely
- High efficiency Linear Technology switching power supplies (good for battery powered applications)
- 50MHz oscillator
- Connector for 1/8 VGA hi-res graphics LCD panel or 16x2 character LCD display
- 60 FPGA I/O's routed to expansion connectors (one high-speed Hirose FX2 connector and four 6-pin headers)
- 8 LEDs, 4-digit seven-segment display, 4 pushbuttons, 8 slide switches

The Nexys provides a reliable "all-in-one" circuit development platform for
engineers at all levels. It can be used to implement designs ranging from
simple l ny other
compon uring a
long ope

The Nex ultipliers
and hig ry
arrays (nd
numero with
embedd

The Nex s,
includin ort and it
ships wi nted
immedia tate of
the art d



Power System
USB, 2.1mm jack, or battery driving 3.3V, 2.5V, & 1.2V switching regulators
Platform Flash Config. ROM
Hirose FX2 100-pin expansion connector
16 Mbytes Flash ROM
16 Mbytes RAM
JTAG
Xilinx Spartan3 XC3SXXX-FT256
(200, 400, and 1000 versions available)
A
B
C
D
8 LEDs
Osc (50MHz)
4 push-buttons
8 slide switches
4 7-seg. displays
6-pin headers

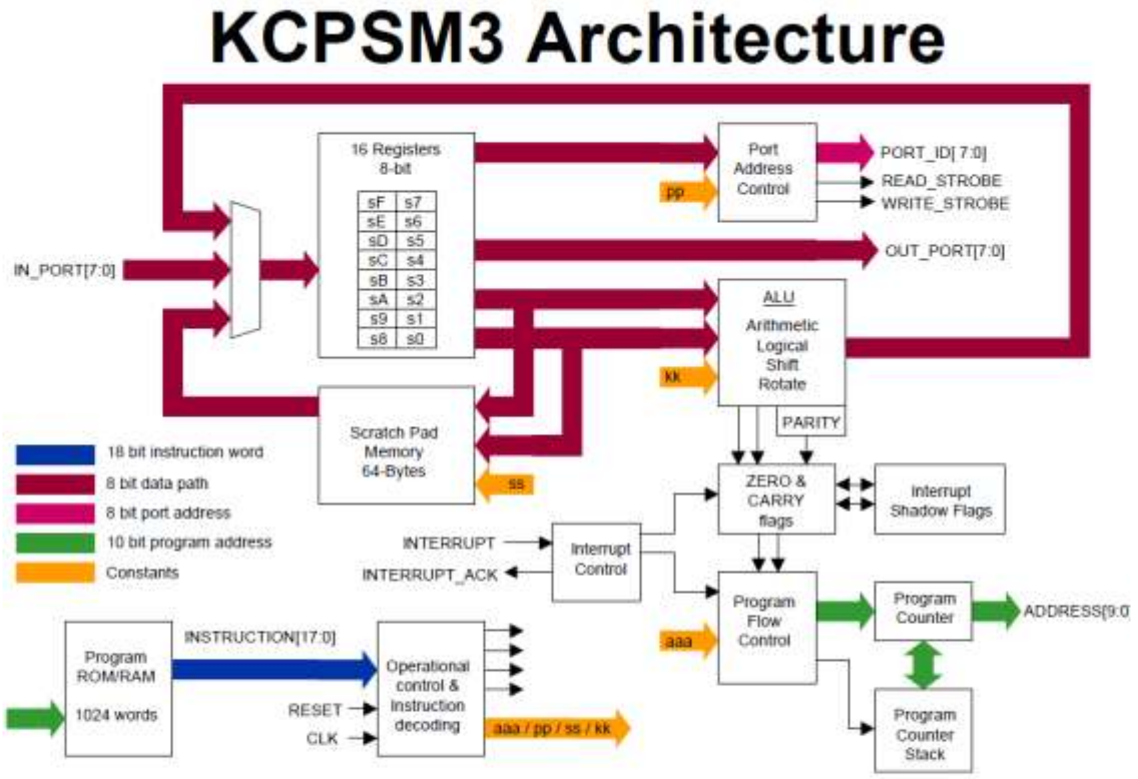
Experimental Setup – Software: Embedded "Picoblaze" Microcontroller (MC)

Picoblaze: (aka "KCPSM3")

- Ken Chapman PSM
- (K) Constant Programmable State Machine (PSM)
- 3rd generation
- 8-bit MC designed for Spartan-3, Virtex-II, & Virtex-II PRO
- 96 slices (5% of available silicon)
- 87 MHz, 43-66 MIPS

Features:

- 16 general purpose registers
- Supports VHDL & Verilog
- Large supporting community



Approach: Literary Review & Gantt Chart

- Digilent Nexys Board
- Digilent Adpet Software
- Xilinx free webpack v12.3
  - Xilinx ISE Design Suite 12.3
  - ISim Simulator
  - MX III Starter (ModelSim)
- KCPSM3.zip (course website)
- PCCOMP – Picoblaze's C Compiler

Tasks		10/18 – 10/24	10/25 – 10/31	11/1 – 11/7	11/8 – 11/14	11/15 – 11/21	11/22 – 11/28	11/30 – 12/6
Study hardware/ software	Nexys I Board	X	🟩					
	XC3S200 FPGA	X	🟩					
	KCPSM3	X	🟩					
Setup development environment	ISE 12.3	X	🟩					
	Adept utility	-	X					
Programming*	Compile simple C-program			1/2	X			
	Peripheral control				3/4	X		
	Embed PicoBlaze					1/2	X	
	SOP example						1/2	3/4
	ADC				-	-		
	Develop metric for speed					-	-	
Deliverables	Presentation							X
	Final report							pending

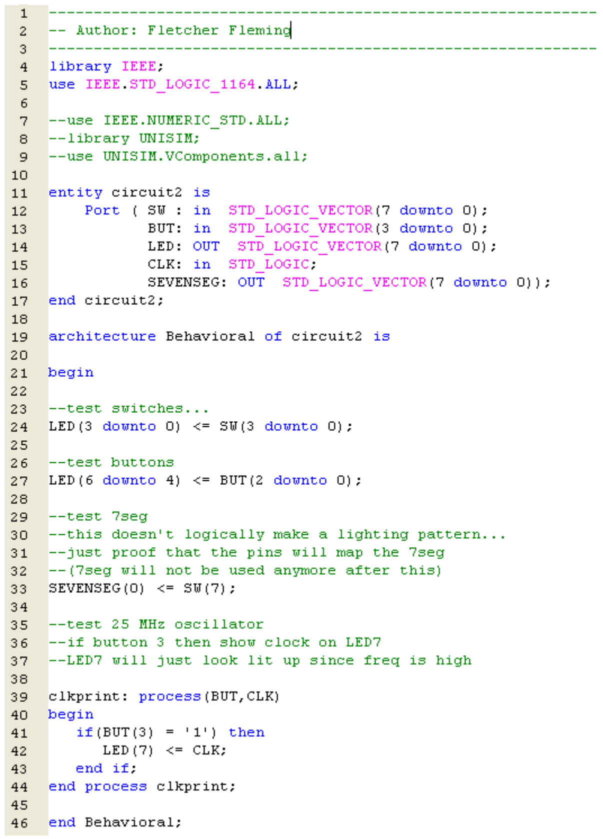
Research and Development: Outline

- Setup development environment
- Completed Nexys & Adept software tutorials
- Achieved peripheral control
  - Required Xilinkx ISE & Digilent Adept
  - Switch, button, LED, oscillator, and 7-SEG controlled
- Completed Picoblaze (KCPSM3) tutorial
  - Including interrupts (not really much to see synthesized)
- Use PCC for a sum of products (SoP) example
  - Either use peripheral or memory to show register values have correct value after SoP
- Investigate analog to digital converters

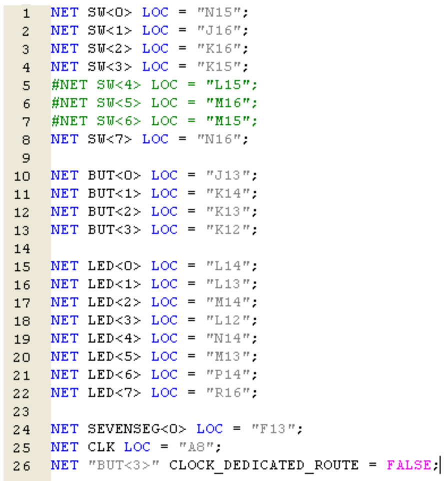
Research and Development: Peripheral Control

Switch, button, LED, oscillator, and 7-SEG controlled

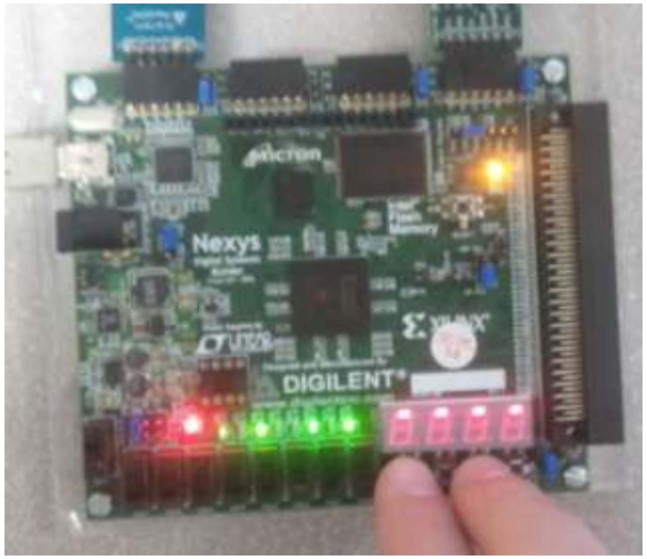
- Useful for debugging



```
----------------------------------------------------------------
-- Author: Fletcher Fleming
----------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

--use IEEE.NUMERIC_STD.ALL;
--library UNISIM;
--use UNISIM.VComponents.all;

entity circuit2 is
    Port ( SW : in  STD_LOGIC_VECTOR(7 downto 0);
           BUT: in  STD_LOGIC_VECTOR(3 downto 0);
           LED: OUT  STD_LOGIC_VECTOR(7 downto 0);
           CLK: in  STD_LOGIC;
           SEVENSEG: OUT  STD_LOGIC_VECTOR(7 downto 0));
end circuit2;

architecture Behavioral of circuit2 is

begin

--test switches...
LED(3 downto 0) <= SW(3 downto 0);

--test buttons
LED(6 downto 4) <= BUT(2 downto 0);

--test 7seg
--this doesn't logically make a lighting pattern...
--just proof that the pins will map the 7seg
--(7seg will not be used anymore after this)
SEVENSEG(0) <= SW(7);

--test 25 MHz oscillator
--if button 3 then show clock on LED7
--LED7 will just look lit up since freq is high

clkprint: process(BUT,CLK)
begin
   if(BUT(3) = '1') then
      LED(7) <= CLK;
   end if;
end process clkprint;

end Behavioral;
```



```
NET SW<0> LOC = "N15";
NET SW<1> LOC = "J16";
NET SW<2> LOC = "K16";
NET SW<3> LOC = "K15";
#NET SW<4> LOC = "L15";
#NET SW<5> LOC = "M16";
#NET SW<6> LOC = "M15";
NET SW<7> LOC = "N16";

NET BUT<0> LOC = "J13";
NET BUT<1> LOC = "K14";
NET BUT<2> LOC = "K13";
NET BUT<3> LOC = "K12";

NET LED<0> LOC = "L14";
NET LED<1> LOC = "L13";
NET LED<2> LOC = "M14";
NET LED<3> LOC = "L12";
NET LED<4> LOC = "N14";
NET LED<5> LOC = "M13";
NET LED<6> LOC = "P14";
NET LED<7> LOC = "R16";

NET SEVENSEG<0> LOC = "F13";
NET CLK LOC = "A8";
NET "BUT<3>" CLOCK_DEDICATED_ROUTE = FALSE;
```



- LED(3 downto 0) <= SW(3 downto 0)
- LED(6 downto 4) <= BUT(2 downto 0)
- 7SEG(0) <= SW(7)
- If(BUT(3) = '1') then

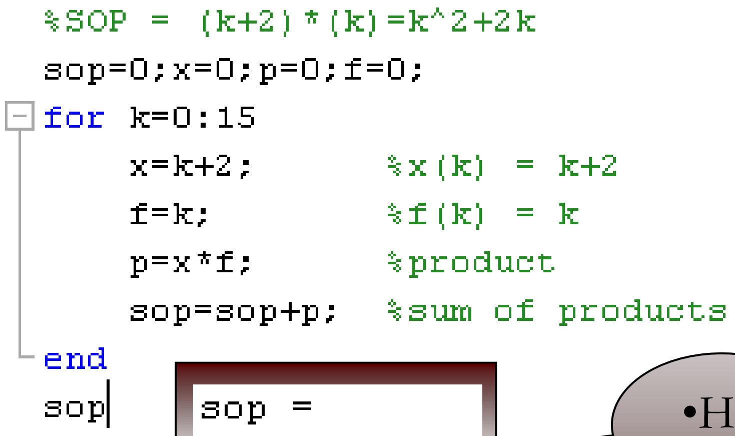
  LED(7) <= CLK

Research and Development: Embedding Picoblaze and PCC

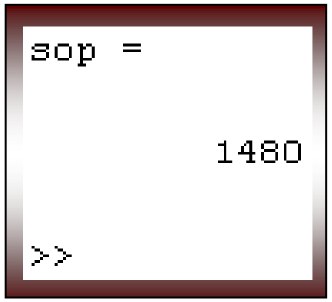
Simple SoP example:

$$\sum_{k=0}^{k=15}(k+2)*k=\sum_{k=0}^{k=15}(k^2+2k)=1480$$

MATLAB Code & output:



```
%SOP = (k+2)*(k)=k^2+2k
sop=0;x=0;p=0;f=0;
for k=0:15
    x=k+2;      %x(k) = k+2
    f=k;        %f(k) = k
    p=x*f;      %product
    sop=sop+p;  %sum of products
end
sop
```



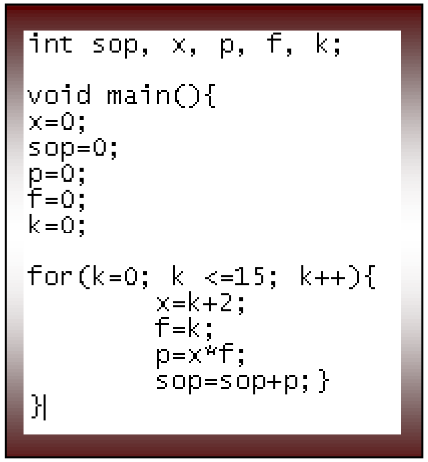
```
sop =

        1480

>>
```

Procedure:

- pccomp -c prog.c
- KCPSM3 prog.asm > out.txt
- Compile prog.vhd via ISE
- Load prog.bit via Adept

•Have not developed speed metric
•Need to show memory first

"C" Code



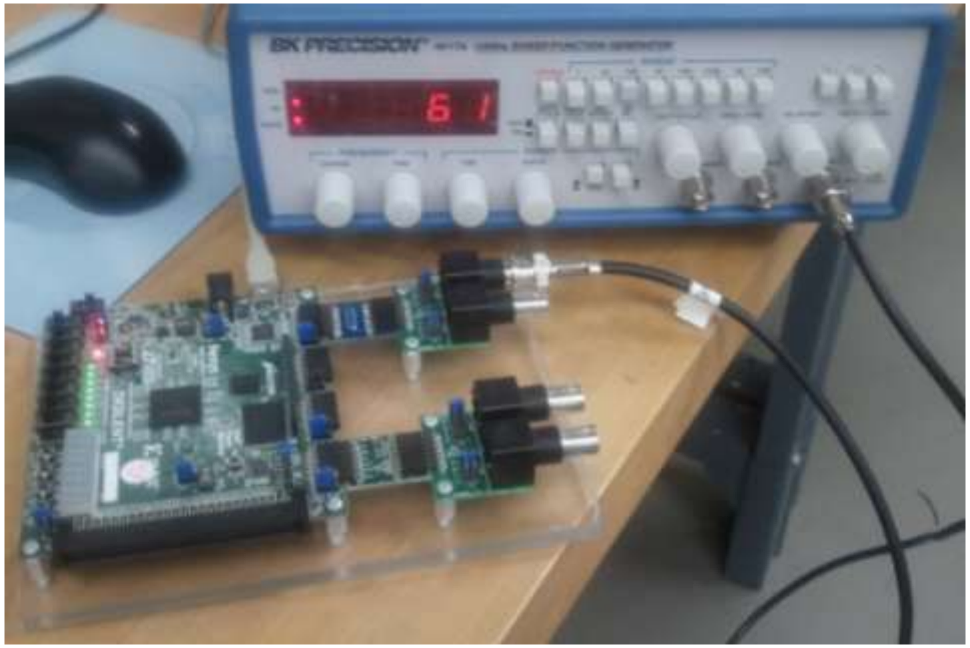
```
int sop, x, p, f, k;

void main(){
x=0;
sop=0;
p=0;
f=0;
k=0;

for(k=0; k <=15; k++){
        x=k+2;
        f=k;
        p=x*f;
        sop=sop+p;}
}
```

Analog to Digital Modules

- Goal was to gain experience prior to using the NiosII (Altera) to assist with sensory input for personal research
- ADC will not be implemented

Stopping point:

- Testing with 60Hz signal
- Goal was similar to HW4 Power Quality
- Need memory dump first for debugging purposes



Limitations and Suggested Future Works

- SoP verification via ModelSim or ISim
  - Show that 1480 is in memory after SoP
- DATA2MEM allows memory changes to occur in bit file without recompile
  - Develop a nice demo to showcase usefulness
- ADC development
  - UART tutorials
- Further investigate and demo the use of interrupts
  - Already completed basic tutorials of this
- Utilize more advanced features of PicoBlaze

  (also write a more complicated "C" program than SoP)

Intended Research Extrapolations

- Test bed development: FPGA control of an electric drive system
  - Input:
    - Various analog sensor measurements (voltage and current)
    - Encoder speed reading of machine
  - Output:
    - Pulse logic to IGBT gate driver card
    - DC voltage reference to DC motor controller
  - The system configuration and control algorithms are the novel study here
  - The FPGA is just the optimal control solution

- An FPGA could run high speed nonlinear models (ex. machines) for controller references
  - It is necessary to investigate methods to solve the underlying equations!
  - Is a softcore processor a viable non real-time solution for solving these differential equations?
  - What simplifications, optimizations, changing in methodology, etc. are needed to approach a real-time solution?

Conclusion

- Showcase memory to prove SoP works properly
- Abandon ADC for time constraints
- Learned (a lot):
  - Xilinx ISE, ISim
  - KCPSM3
  - PCC
  - Adept
  - Scheduling, independent software based research, and a lot about embedded processors