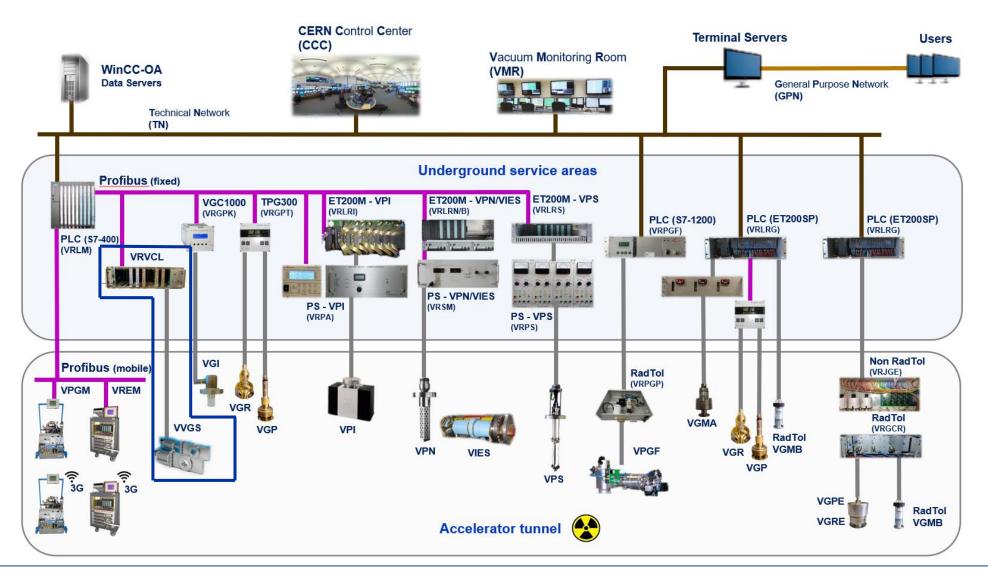


New Sector Valve Control Unit

Sara M. G. Soares

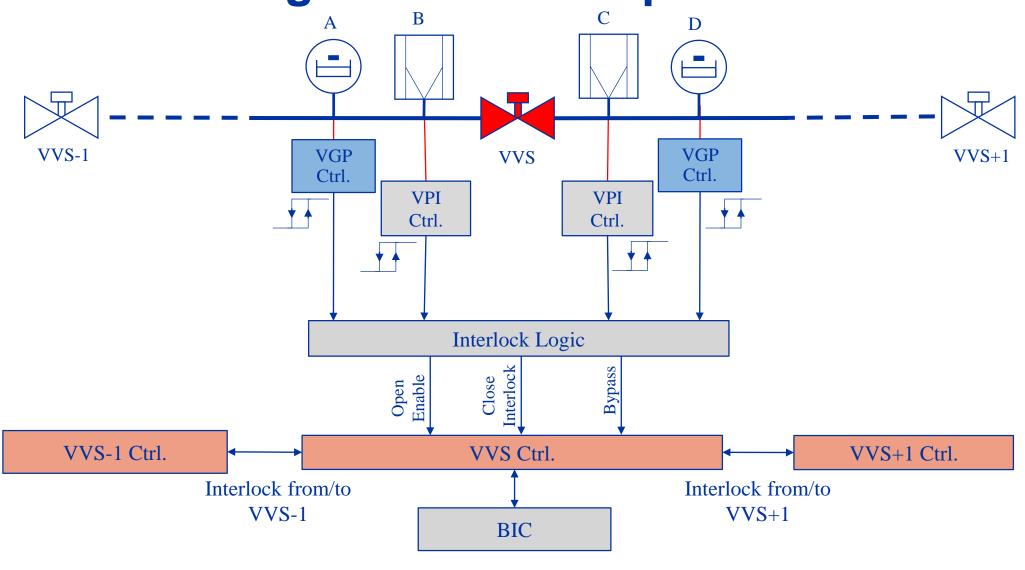
6/12/2022

Vacuum Control System Architecture



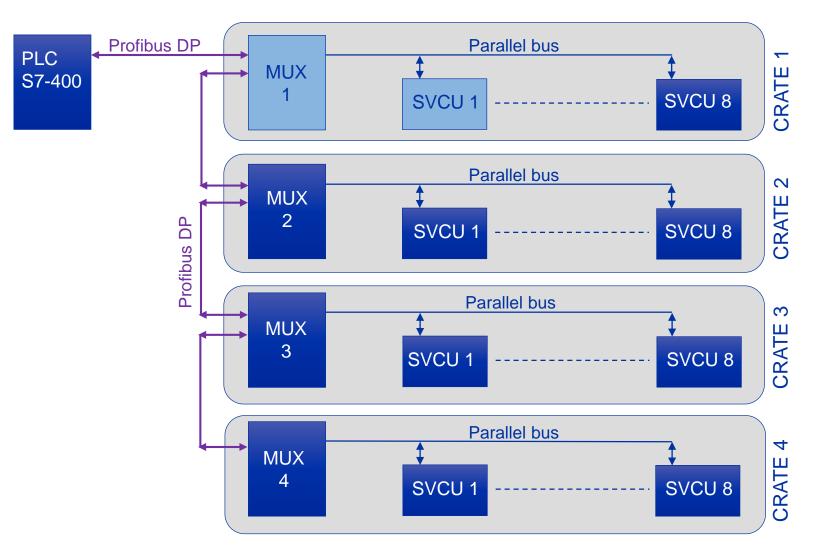


VVS Interlock logic – LHC example





Sector Valve Controller Architecture





Programmable components



- CPLD (Complex Programmable Logic Device)
- Internally based on LUTs (Look-Up Tables)
- Contain embedded flash memory
- Smaller timing delays (compared to FPGAs)



FPGA (Field Programmable Gate Array)

- Internally based on CLBs (Configuration Logic Block)
- Flexible
- Timing delays depend on implementation method

- Programmable Logic Device family (PLD)
 - Programming a PLD consists of specifying how the logic gates will be interconnected upon power-up.
 - Inherent parallelism (fast and predictable reaction time)
 - No instructions are executed



Programmable components



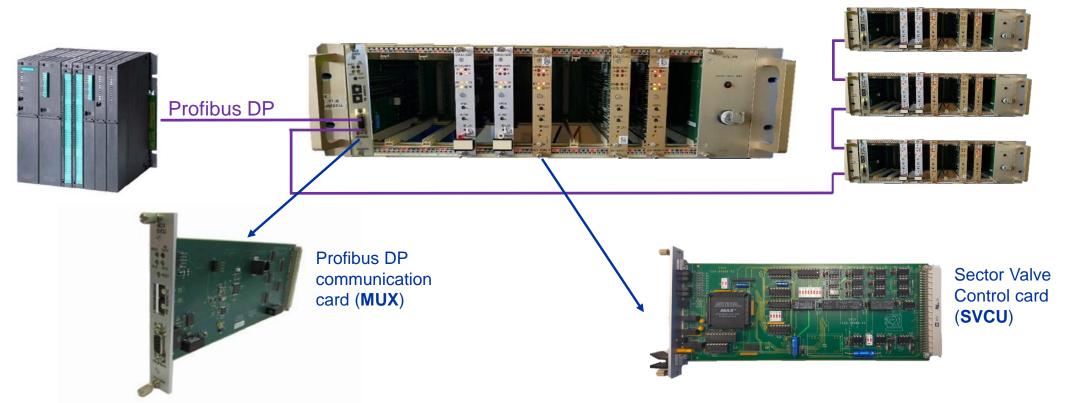
Microcontroller (MCU)

- Computing system
- Designed for embedded application
- Digital I/Os
- DAC and ADC

- Programming a microcontroller consists of defining a list of instructions stored within its flash memory which are then executed sequentially after power-up.
- Relatively slow (compared to CPLD and FPGA)
- No parallelism



Sector Valve Controller



- Existing Version (in production)
 - Based on microcontroller 18F6527 from Microchip
 - Profibus slave ASIC (Application Specific Integrated Circuits) VPC3+S from profichip

- Existing Version (in production)
 - Based on CPLD (Complex Programmable Logic Device) MAX from Altera



New Design Motivations

- SVCU:
 - 20 years old design
 - CPLD not available anymore on the market
 - Quartus Altera software not compatible anymore
 - Logic described graphically and only with combinational logic
 - No Clock system
 - Obsolescence of other components
 - Only compatible with parallel backplane bus communication
 - No protection against surges of voltage/current
- MUX:
 - Only compatible with parallel backplane bus communication
 - Must integrate the functionalities of the new SVCU card
 - No protection against surges of voltage/current







New Design Specifications

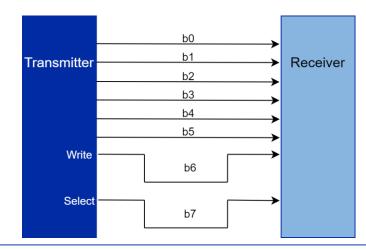
- Keep the existing crate and backplane
- Backward compatible with the current system for smooth deployment
- Implement SPI (Serial Peripheral Interface) serial communication with differential lines (MUX + SVCU):
 - Aim to replace the current parallel communication
 - Improved reliability (less sensitive to glitches)
 - No limitation of data transmission (not limited by the number of lines of the parallel bus)
- Use FPGA and clock system for sequential logic and timing applications (SVCU)
 - Use of VHDL language for logic description
- Choose components with extended life on the market (MUX + SVCU)
- Improve I/Os overcurrent, overvoltage and ESD (Electrostatic Discharge) protection (MUX + SVCU)



Communication Protocols – Parallel Vs SPI

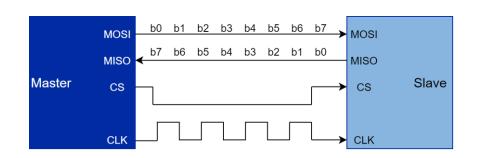
Parallel Communication

- Length of message depends on number of wires
 - Limited by the protocol and number of lines
- Sensitive to noise/glitches



SPI communication

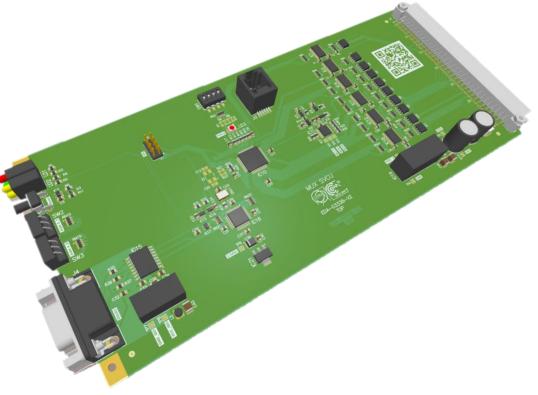
- Arbitrary length of message
 - Not limited by the number of lines
- Faster transmission rate
- More robust against noise/glitches





MUX card design

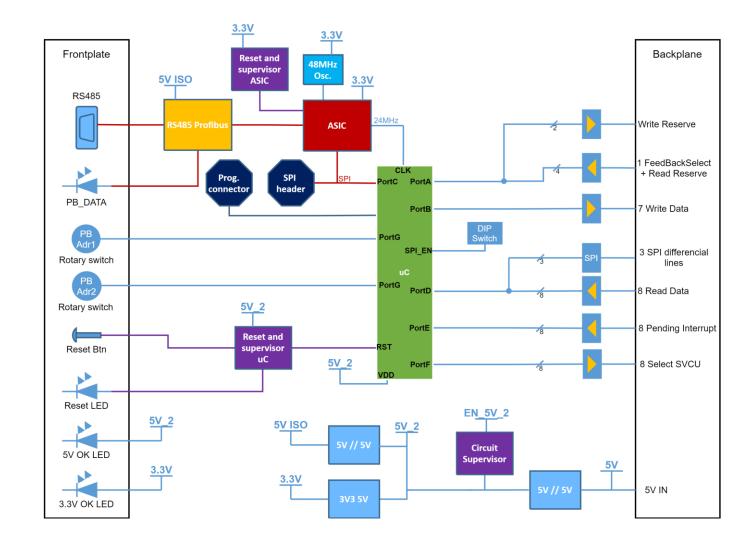
- Prototype of MUX card in production
 - New components
 - New buffers with output enable
 - New voltage supervisors
 - Improved protection against surges of current and voltage
 - TVS (Transient Voltage Suppression) diode arrays added to all backplane connection lines
 - SPI hardware added
 - Differential buffer with output enable





MUX card design - Hardware

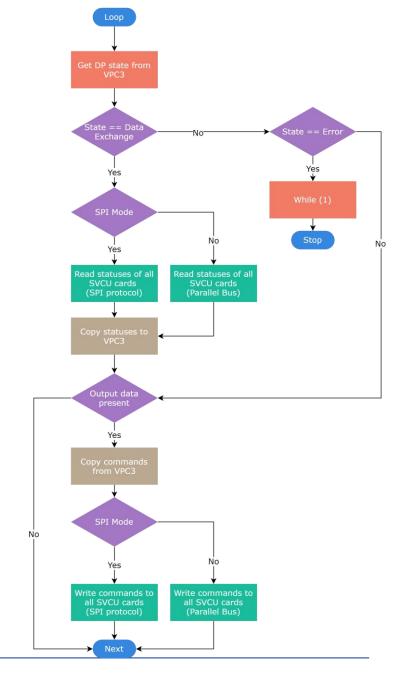
- PIC18F6527 microcontroller
 - 8-bit microcontroller
 - SPI compatible
- Profibus slave ASIC VPC3+S
 - Provides 24 MHz clock to the microcontroller





MUX card design - Firmware

- Programmed using C
- Program in a loop
- Checks state of the ASIC chip
- Depending on the Communication mode, it follows the SPI protocol or the Parallel bus protocol
- If program detects an error, it enters an infinite loop, and stops after the WatchDog timer is triggered





SVCU card design

- Prototype in production
 - New components
 - New buffers with output enable
 - New voltage supervisors
 - SPI hardware added
 - Differential buffer with output enable
 - Added protection against surges of current and voltage
 - TVS diode arrays added to all backplane connection lines
 - Optimized circuit logic and behavior
 - Optimized logic for enabling of SPI/PRL components
 - Added pull-up/down resistors to fix signals at start-up to avoid errors

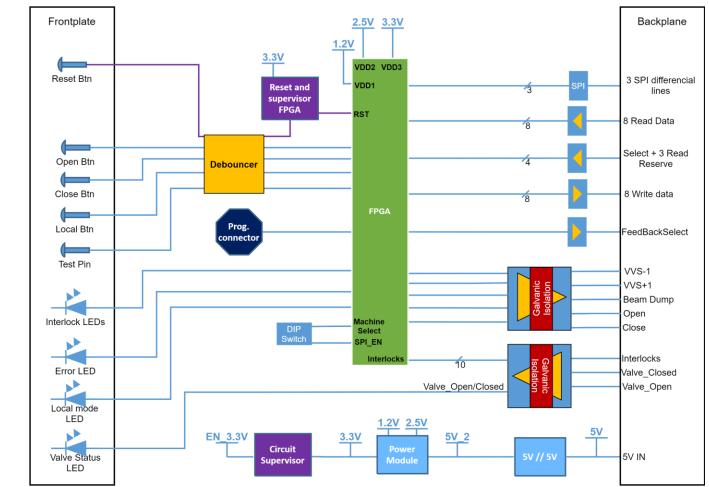




SVCU card design - Hardware

IGLOO2 FPGA

- On-chip 1 MHz oscillator
- 60 dedicated I/O pins
- 84 user I/O pins
- Galvanic isolation
 - Optocouplers for interlocks and valve actuation
 - Relays for Beam Dump and Valve status
- Dip Switch to control SPI or Parallel mode
- Dip Switch to select which Machine it operates in





Gateware – VHDL for SVCU card

	Top_Module_0 & X StatusLED.vhd	ParalelCommunication.vhd 5 × Remote Local Sw	atching.vhd العامية المعامية ا
dule(root): Top_Module_0	e = ∋ A+ A ✓		
Herarchy Annual Annua		tite lines to make sure that no glitches	s or sudden spikes in the signals
	29 affect the operation of the 30	system.	
work XTLOSC_FAB(DEF_ARCH) (osc_comps.vhd) [work]	31 L		
TLOSC(DEF_ARCH) (osc_comps.vhd) [work]	32 library IEEE; 33 use IEEE.std logic 1164.ALL;		
Top_Module_0	34		
SD Top, Module, 0 FIGA Testing FIGA Testing HardwareInterface(architecture_HardwareInterface) (HardwareInterface	35 E Uncomment the following lib.	ary declaration if using	
E Dp_core	36 L arithmetic functions with 5	gned or Unsigned values	
ParallelCommunication(Behavioral) (ParallelCommunication.vhd) [37 use IEEE.NUMERIC_STD.ALL; 38		
Remote_Local_Switching(arch) (Remote_Local_Switching.vhd) [work] StatusLED(Behavioral) (StatusLED.vhd) [work]	39 Mentity ParallelCommunication i.		
ValveActuation(Behavioral) (ValveActuation.vhd) [work]	40		
spi_slave(rtl) (spi_slave.vhd) [work]	41 印 Fort (
test_component/architecture_test_component) (test_component.vhd) [work]	42 43 INPUTS		
User HDL Source Files	44 clk	; in std logic;	Clock signal
Components	45 reset	: in std_logic;	Reset signal
	46		
	47 INTRLCK		Interlock signals vector. Contains the inputs from all interlocks
	48 DIN 49	: in std_logic_vector(7 downto 0);	Read Commands buffer. Stores the information sent by the PLC
	50 5W0	: in std logic;	
	51 SWC	: in std_logic;	
	52 VS	: in std_logic;	Valve Status signal. If '0', Valve is CONNECTED
	53 EXT 54 REMT	: in std_logic;	Local Temp. Interlock. If '0', Interlock OK
	54 REMT 55 T PIN	: in std_logic; : in std logic;	Remote Control signal. If '1', Local Control Mode; If '0', Remote Control Mode Test Pin inserted signal. If '0', Test Pin is INSERTED
	56 BDR	: in std logic;	Beam Dump Request signal. If '0', Beam Dump REQUESTED
	57 MachineSEL		Machine Select signal. Switches between the 3 machines (LHC, SPS, CPS)
	58		
	59 CUTPUTS 60 DOUT		
	60 DOUT 61 PRL CMD	: out std_logic_vector(5 downto 0) : out std logic;	; Kead Status sent to the FLC - signal to change from local to remote mode (parallel)
	62 BACK		 - signal to change from local to remote mode (parallel) - Back status signal. It serves as a sort of acknowledge signal to assure correct operation
	63 CLOSE REM		- Internal signal to ValveActuation block
	64 OPEN_REM		ternal signal to ValveActuation block
	65 -);		
	66 end ParallelCommunication; 67		
	68 L		
	69 Parchitecture Behavioral of Par	ilelCommunication is	
	70 71 WRITE REGISTER		
	72		
W Design Herarchy Stimulus Herarchy Files HDL Templates Catalog			
sages 🔞 Errors 🗼 Warnings 🌒 Info			
ng file 'Ci\git\SvCU\FFGA_ALL_HACHINES\ndi\spi_slave.vnd'.			
ng file 'C:\git\SVCU\FPGA_ALL_MACHINES\hdl\StatusLED.vhd'.			
ng file 'C:\git\SVCU\FPGA ALL MACHINES\hdl\test_component. ng file 'C:\git\SVCU\FPGA ALL MACHINES\hdl\UserLogic test.	vhd.		
ng file 'C:\git\SVCU\FPGA ALL MACHINES\hdl\ValveActuation.	vhd'.		
ng file 'C:\git\SVCU\FPGA_ALL_MACHINES\stimulus\Top_Module			
PGA_TEST project was opened.			
Log Search Results Cores			

 Programmed using VHDL (Very High-Speed Integrated Circuit Hardware Description Language)

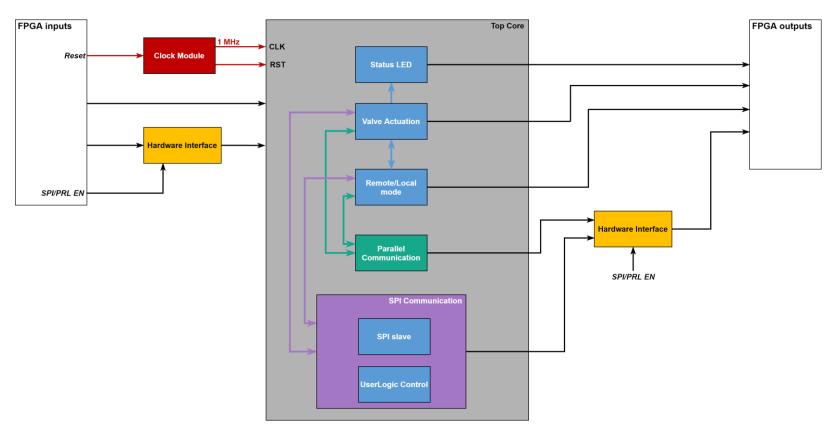
 Microsemi (Microchip) software used – Libero IDE (Integrated Development Environment)

• Supports ModelSim (Simulation Tool)



Gateware – VHDL functional blocks

- 5 Function blocks
- 1 MHz clock signal generated internally by the FPGA
- Hardware Interface block to translate signals for the different blocks:
 - Separation of SPI/PRL lines based on communication mode





Validation of the VHDL code and Prototypes

- VHDL code first simulated using ModelSim
- Pre-Synthesis simulation Validates the logic of the written code
- Post-Synthesis simulation Code is converted into a logic circuit and simulated.
 Validates the generated circuit

/SYSCLK 0	0	JUUL	hnn	ww	JUUUU	MMM	սուղ	սու	ուսով	MMM	MMM	MMM	MM	NNN	innn	վույու	MMM	MM	www	սոոս	MMM	տուր	WW	MMM	MMM	տուվու	տվուտ	JUUUL
/NSYSRESET 0	0																											
/KEY 1																												
/BLR 1																						h μ						
/testing_top_0/Remote_Local_Switching_0/BLR1 1																												
/testing_top_0/Remote_Local_Switching_0/BLR2 1								$\neg \downarrow$																				
/testing_top_0/Remote_Local_Switching_0/BLR3_0																				ſ			ſ					
/testing_top_0/Remote_Local_Switching_0/state F	RM	RM	<u>) (</u> R	м) (RI	М		(LM	(RM (LI	1			(RM) LM	(RM		<u>) (rm</u>) (RM			<u>) lm (ri</u>	M (LM	(RM		
/REMT 0	0															↑												
/SPI_CMD 0																												
/PRL_CMD 0	0																											



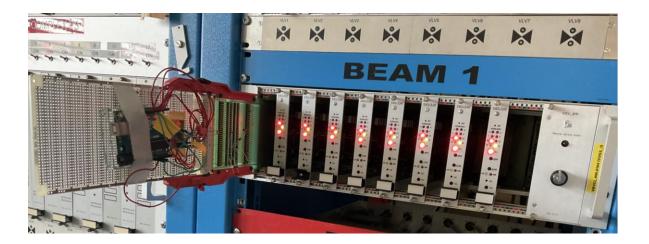
Validation of the VHDL code and Prototypes

Laboratory testbench to test prototype cards

MUX card used to test Parallel Communication

Arduino used to test SPI communication



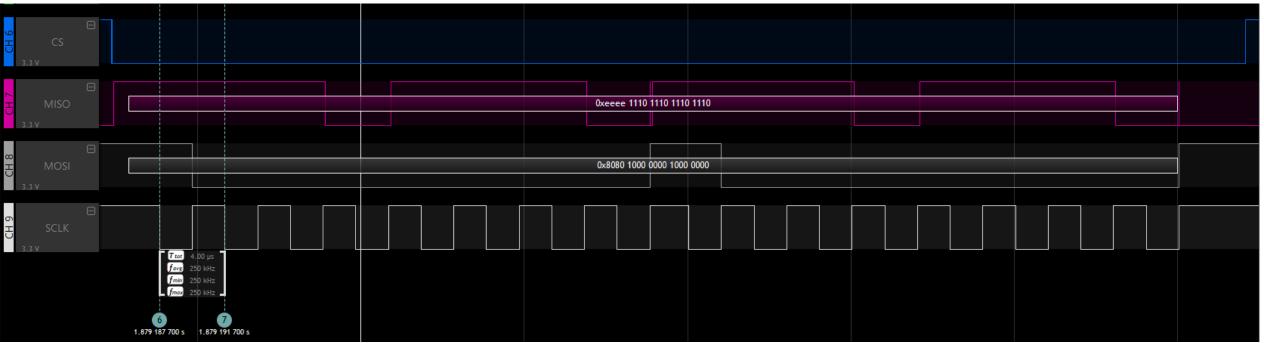




SPI Communication - Bus analyzer

• Use of digital bus analyzer to validate SPI messages



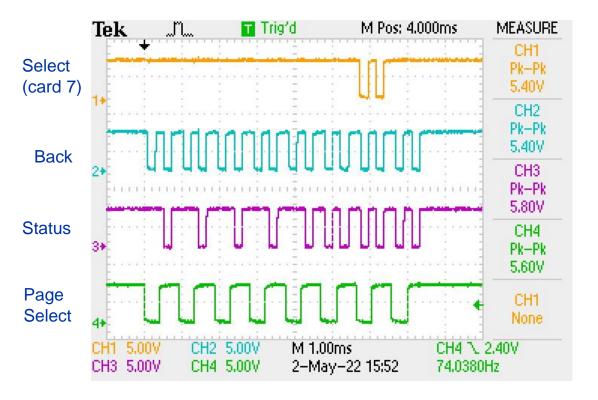




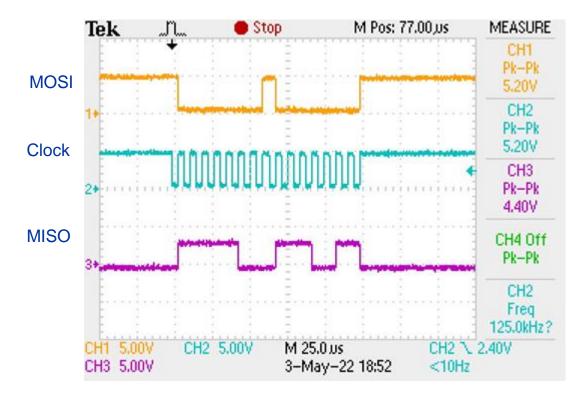
Validation of the VHDL code and Prototypes

Laboratory testbench to test the prototype cards

Parallel Communication



SPI communication



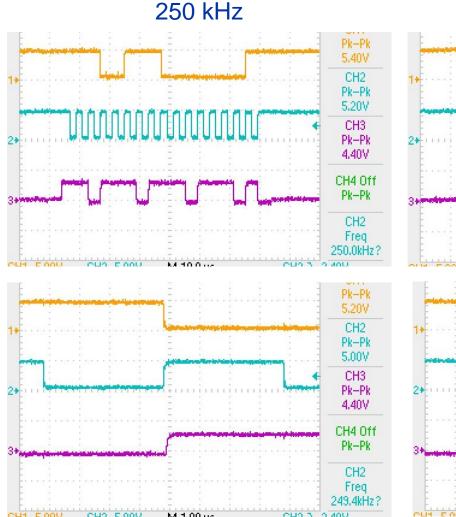


SPI transmission speed

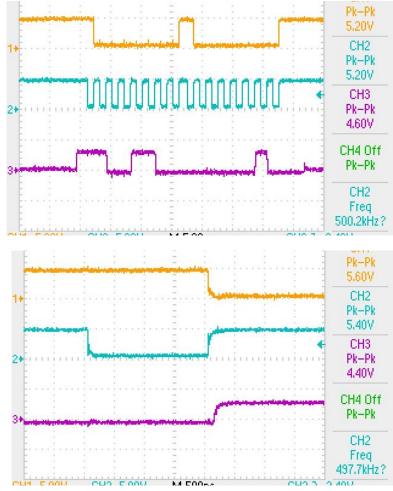
• Max SPI frequency:

 $\frac{SYSCLK}{2} = \frac{1 MHz}{2} = 500 kHz$

- Higher frequency = More distortion
- 3 Different frequencies tested:
 - 125 kHz
 - 250 kHz
 - 500 kHz Max frequency



500 kHz





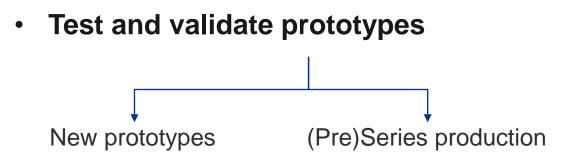
Future Work

SVCU

- Produce prototypes
- Review VHDL code and rewrite it to fit
 new prototype
- Test and validate prototypes
 New prototypes (Pre)Series production

MUX

- Produce prototypes
- Review C code and rewrite it to fit new prototype







Thank you!