ELEC 4200 Lab#10
Interrupting SPI Receiver for use with a Processor Core

AUBURN UNIVERSITY

SAMUEL GINN COLLEGE OF ENGINEERING

- SPI description:
http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus

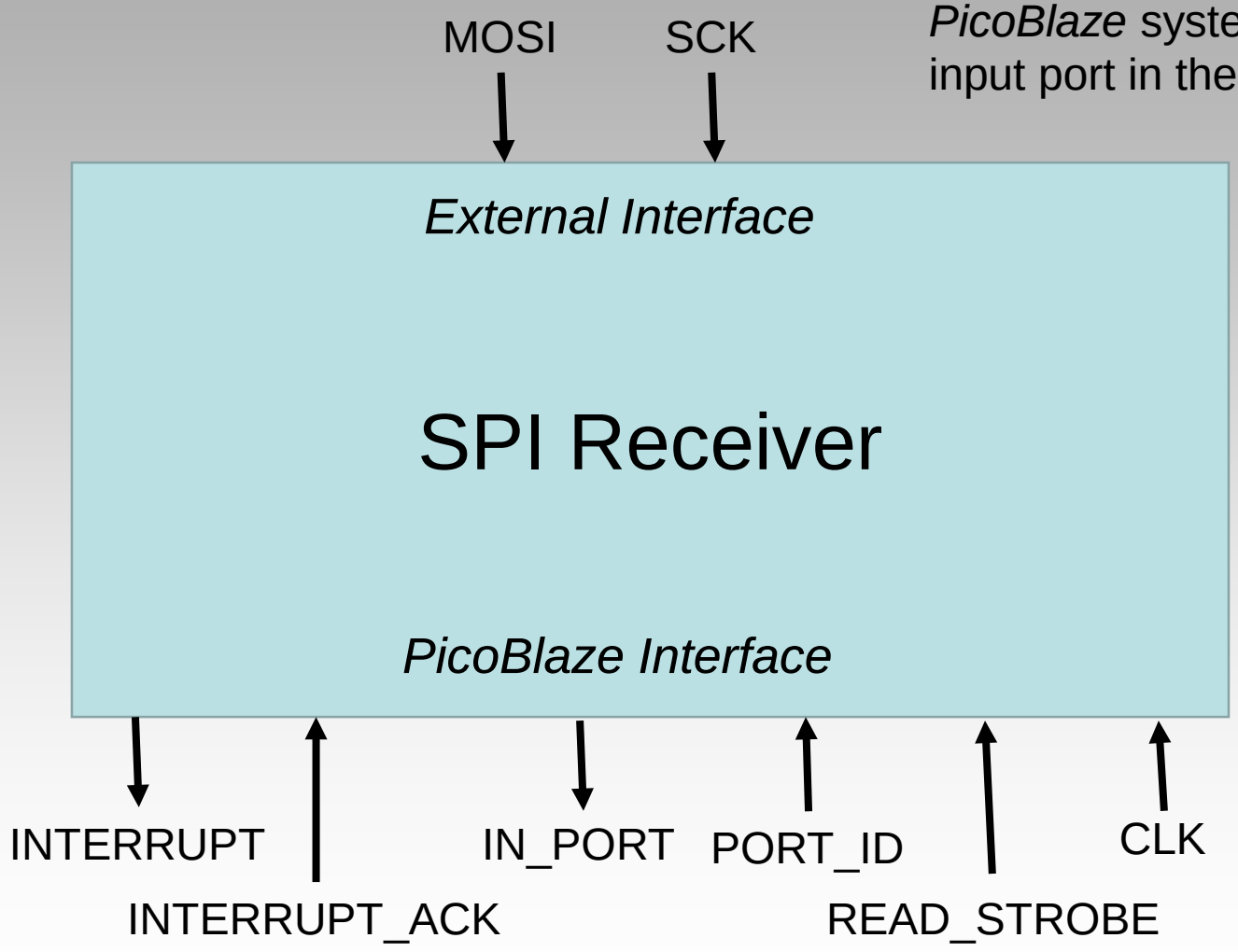
- References you may need:
  - *PicoBlaze KCPSM6 User Manual*
  - *PicoBlaze 8-bit Embedded Microcontroller User Guide*

Overview

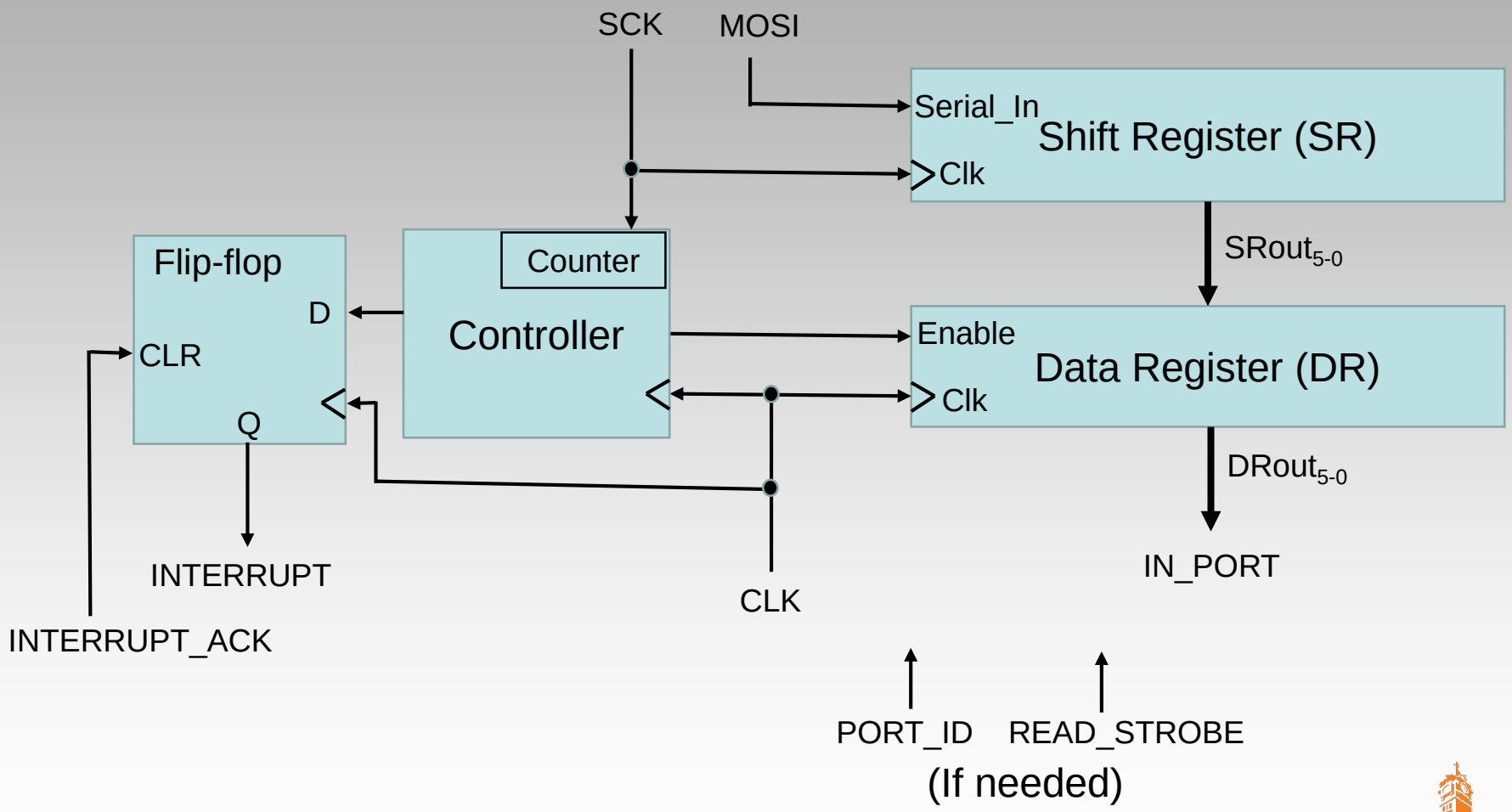
- Design a ***Serial Peripheral Interface (SPI)*** ***<u>receiver</u>*** that can be interfaced (later) to *PicoBlaze* as an interrupting input port
  - » Requires basic understanding of SPI operation
  - » Requires understanding of *PicoBlaze* input/output ports and interrupt mechanism
- The SPI receiver is to operate as a “slave device” on the NEXYS 4 DDR board
  - – Serial data is to comprise 6 bits of information
  - – SCK (shift clock) is to be supplied by a push button
  - – MOSI (master-out/slave in) serial data is to be supplied by a slide switch or push button
- The SPI receiver is to activate an INTERRUPT signal when new data is available for *PicoBlaze* to read
- The SPI receiver should operate as a *PicoBlaze* input port, with received data read by an INPUT instruction

SPI Receiver Module

To be instantiated in the *PicoBlaze* system as an input port in the final project.



SPI Receiver Structure



SPI Receiver Operation

- MOSI value is to be shifted into SR on each rising edge of SCK
  - Data is supplied on MOSI least significant bit first
- On the next rising edge of CLK after 6 bits have been shifted into SR:
  - Transfer the data from SR -> DR
  - Activate the INTERRUPT signal (to indicate data available)
- When *PicoBlaze* activates INTERRUPT_ACK, reset the INTERRUPT signal
- *PicoBlaze* should read the data from DR via an INPUT instruction
  - The receiver operates as an "input port"
  - Review timing of PORT_ID, IN_PORT, READ_STROBE, CLK

- **Optional:** detect "overrun" errors
  - Set a "flag", DA, when data is transferred from SR -> DR
    - » DA=1 indicates unread data available in DR
    - » DA could be tested by software, if interrupts are not desired
  - Reset the DA flag to 0 when DR is read by an INPUT instruction
    - » Indicates data has been read
  - Set "overrun error" flag, OE, if new data is transferred from SR -> DR while the DA flag is set

SPI Receiver Design

VHDL Model Specifications

- Entity inputs/outputs as shown on Slide 3
- Behavior as described on Slide 5
- Structure as shown on Slide 4. Write as one VHDL model, using a separate process for each of the major components
  - Shift register SR
  - Data register DR
  - Interrupt flag
  - Controller (you may use more than one process for this, if desired)
  - SCK debounce
    - Since a push button will be used for SCK, “debounce” it to ensure that only one bit is captured per button press, perhaps using something like the “one shot” circuit from previous labs
- Use switches and LEDs on the NEXYS 4 DDR board for testing the SPI receiver
  - Buttons/switches for MOSI, SCK, CLK, and INTERRUPT_ACK
  - LEDs to show states of INTERRUPT and IN_PORT

Pre-lab Assignment

- Review the class notes on SPI operation
- Review the *PicoBlaze KCPSM6 User Manual* sections on:
  - INPUT/OUTPUT port design
  - INTERRUPT and INTERRUPT_ACK signals
- Write a VHDL model of the SPI receiver
  - Be prepared to verify this design via simulation
- Design a test procedure to verify the SPI receiver
  - Supply serial data via MOSI/SCK (multiple 6-bit values)
  - View received data and INTERRUPT signal
  - Supply a CPU clock
  - Signal an INTERRUPT_ACK, as would be generated by *PicoBlaze*

Lab Exercise

- Simulate your SPI receiver VHDL model to verify its correctness.
  - Verify the SPI receiver model in Active-HDL
  - Verify the PicoBlaze system containing the SPI receiver
- Map the SPI receiver input/output signals to buttons/switches and LEDs on the NEXYS 4 DDR board.
- Build (synthesize, map, place, route) and download the SPI receiver to the NEXYS 4 DDR board
- Demonstrate the operation of the SPI receiver to the GTA

Report Guidelines

- Be sure to include all sections required by the lab manual guidelines. In addition be sure your report includes the following:
  - SPI receiver VHDL model
  - Simulation procedure and results
  - Steps taken to simulate, synthesize, and download your code
  - Synthesis results (LUTs, FFs, slices, etc)
  - Experimental results - what went right and wrong in your design