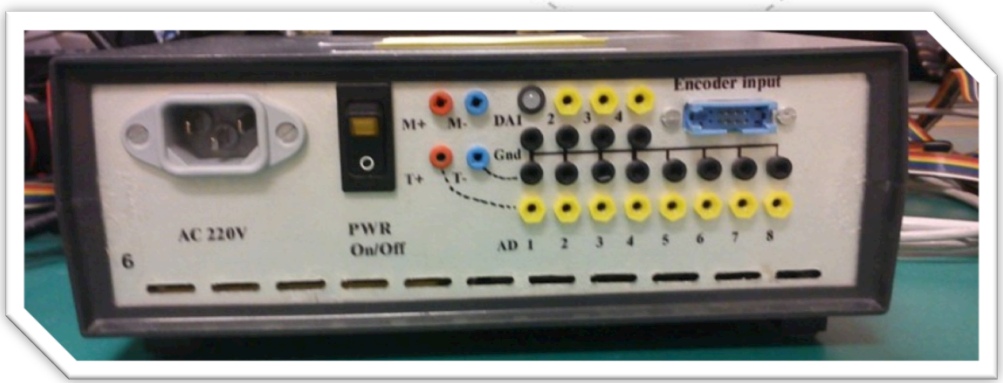
TTK4155

Industrial and Embedded Computer Systems Design

NTNU – Trondheim
Norwegian University of
Science and Technology

Lab lecture 7

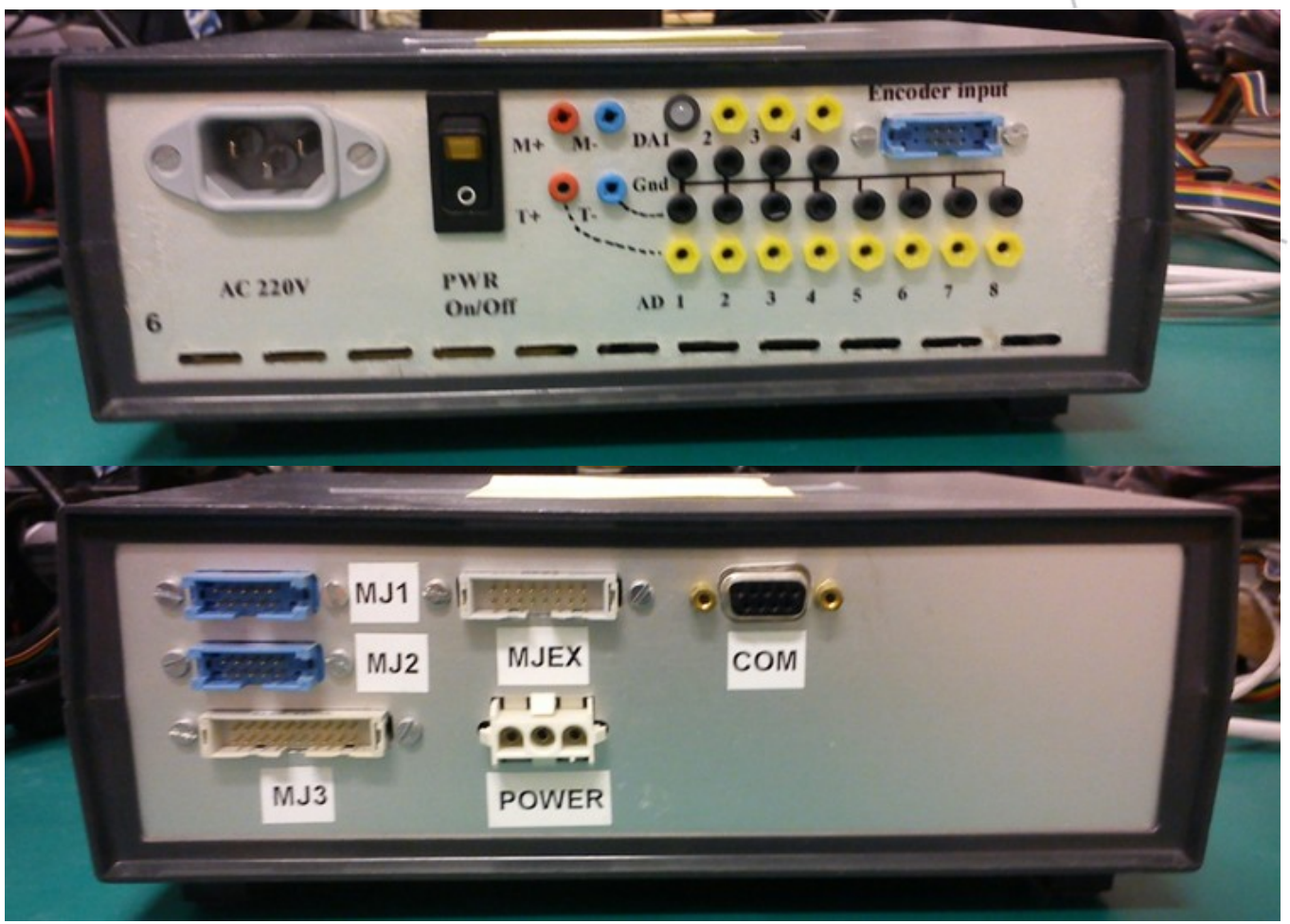
- I2C bus
- Motor controller box
- Solenoid



Exercise 7: Controlling motor and solenoid

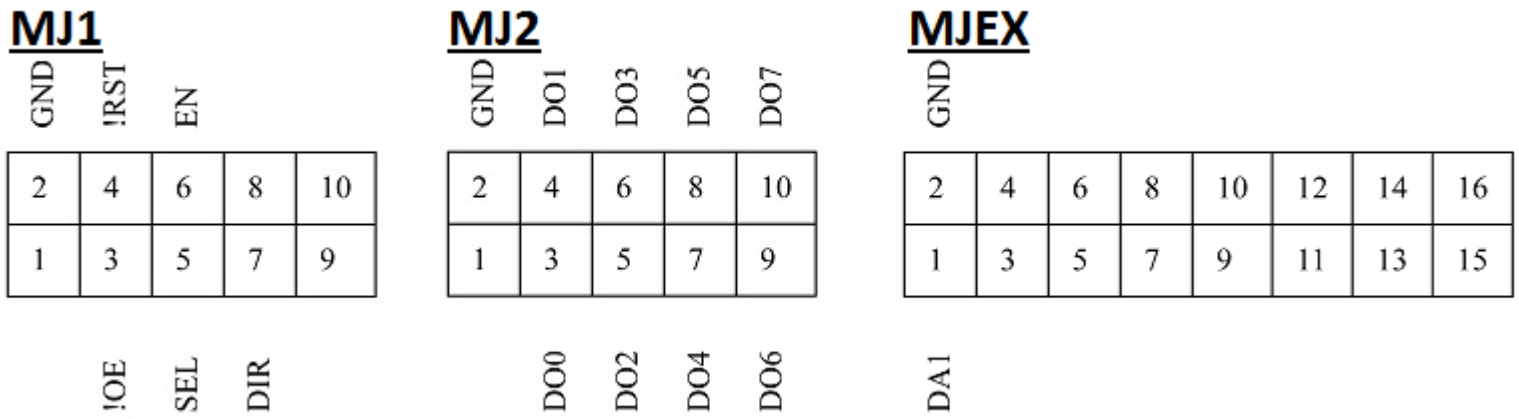
- In this exercise, you will
  - Connect the motor controller box to STK500 and I/O board
  - Connect the ping pong board to the motor controller box
  - Use Atmel's I2C driver to control the DAC on the I/O board
  - Create a motor control driver and test with joystick input
  - Create a speed and/or position controller using feedback from the quadrature encoder
  - Connect the solenoid and use a button on the joystick to fire

Motor box interface



Motor box interface

- Simple control pin interface



- Connect motor to M+ and M-
- Connect encoder to “Encoder input”

Basic control of motor

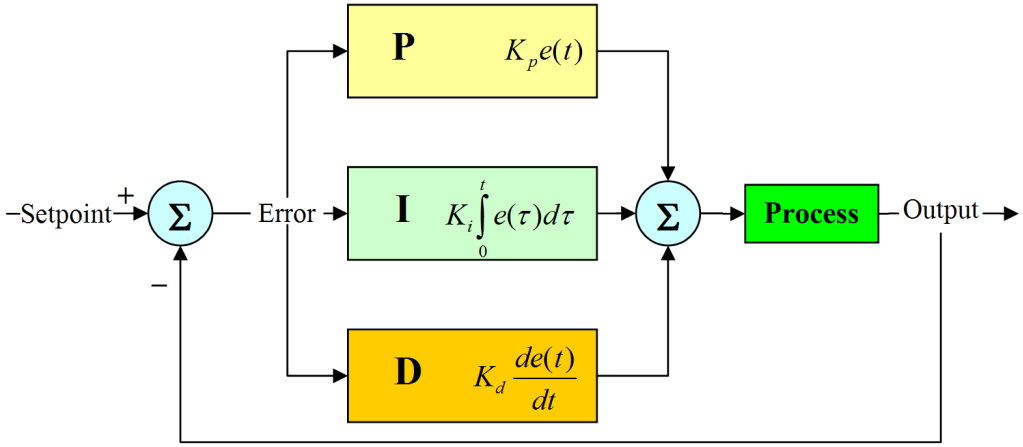
- Set motor “speed” by adjusting DAC voltage
- Select direction with DIR pin
- Set EN pin high to enable motor

Read motor encoder

- Internal 16 bits counter
- !OE pin activates output of encoder counter
- SEL selects either high byte (0) or low byte (1)
- !RST resets the counter

Position regulator

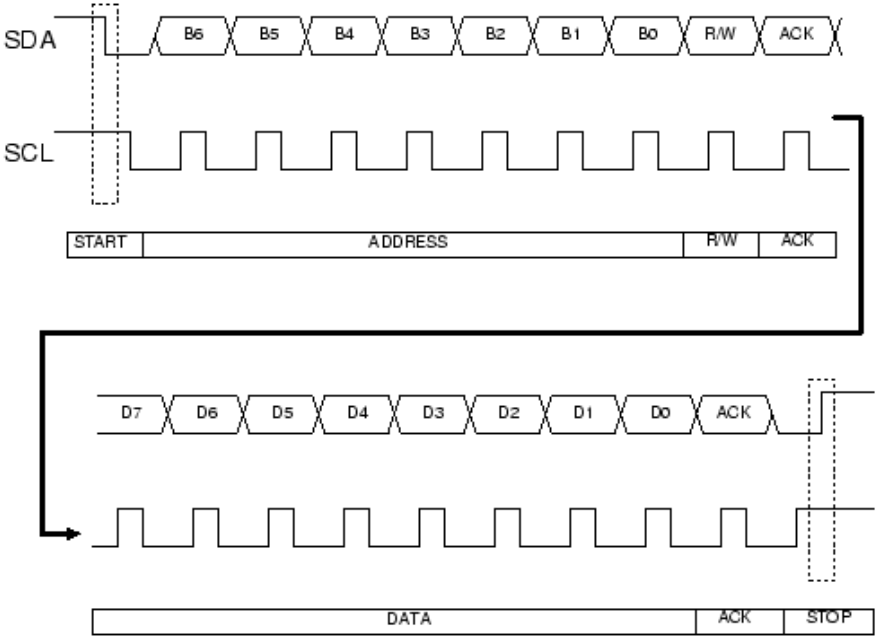
- For example a PID regulator

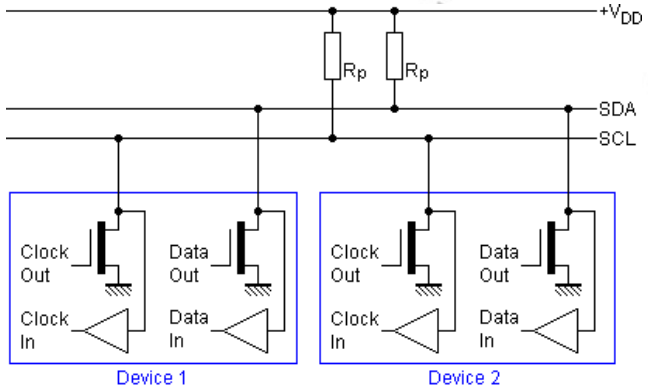


- The quadrature encoder will give an indication of the motor position
- The error is the difference between the indicated position and the wanted position (from the joystick)
- Calculate the necessary values P, I and D
- Sum all values and apply to motor

DAC interface

- Use I2C to interface with the DAC
- I2C is a two-wire protocol, clock line and bi-directional data line
- Master/slave configuration
- Fairly simple protocol:





Atmel I2C driver

- To make things simple you can download the I2C driver provided by Atmel: http://www.atmel.com/dyn/resources/prod_documents/AVR315.zip
- This driver is written for a different compiler (IAR), so some changes have to be done:
  - #include "ioavr.h" → #include <avr/io.h>
    #include "inavr.h“ #include <avr/interrupt.h>
  - #pragma vector=TWI_vect
    __interrupt void TWI_ISR(void) → ISR(TWI_vect)
  - __enable_interrupt(); → sei();
- To use the driver you must
  - Include the twi file
  - Initialize TWI module and enable interrupts
  - Use TWI_Start_Transceiver_With_Data(unsigned char *msg, unsigned char msgSize)
- Also read the appnote: http://www.atmel.com/dyn/resources/prod_documents/doc2564.pdf

Using the MAX520 DAC

- Pretty straight forward. Read the data sheet

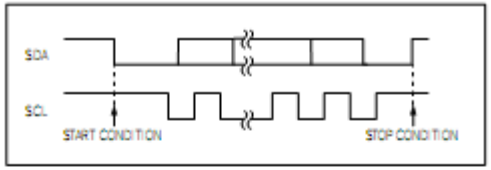


Figure 4. All communications begin with a START condition and end with a STOP condition, both generated by a bus master.

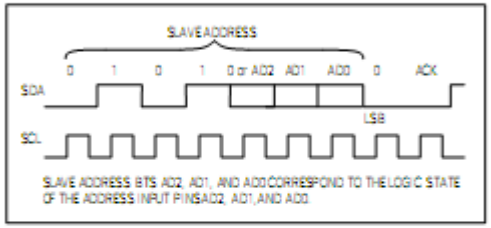


Figure 5. Address Byte

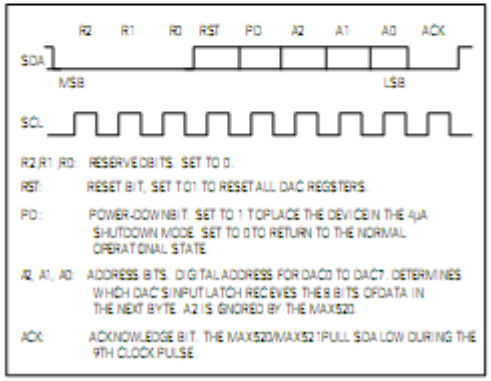
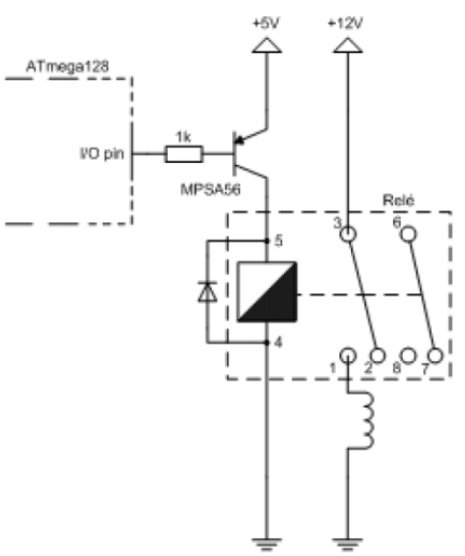


Figure 6. Command Byte

NTNU – Trondheim
Norwegian University of Science and Technology

Solenoid

- The solenoid requires 12V (at least) to throw a punch
- A relay have to be used
- But the AVR I/O pins can not supply the necessary current to drive the relay
- A transistor is used
- Protective diode at the relay input



Questions?