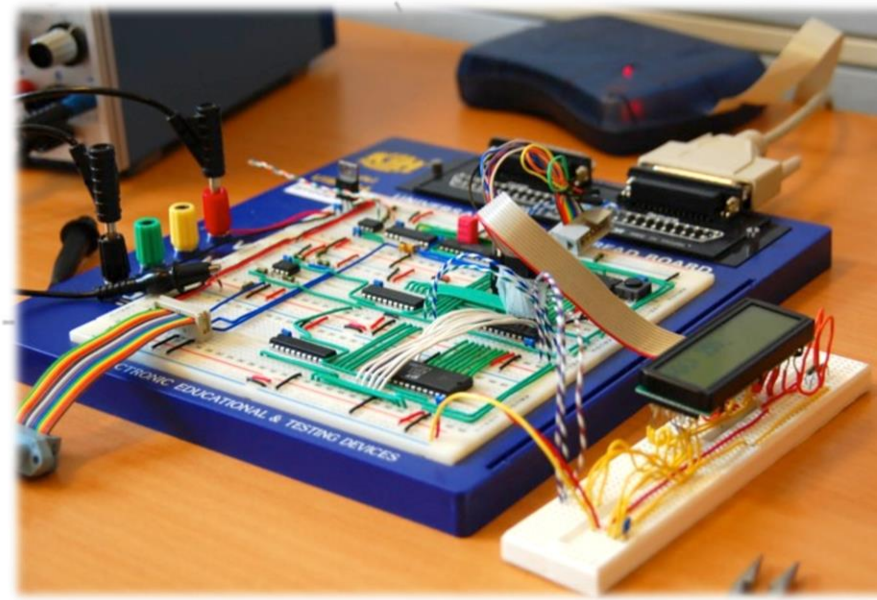
TTK4155

Industrial and Embedded Computer Systems Design

NTNU – Trondheim
Norwegian University of Science and Technology



Lab lecture 1

- Introduction to the term project
- General and practical information
- Exercise1
- Lab group signup

Lab team

- Waseem Hassan
  - waseem.hassan@ntnu.no or D351A
  - Administrative tasks (groups, lab etc.), lab lectures

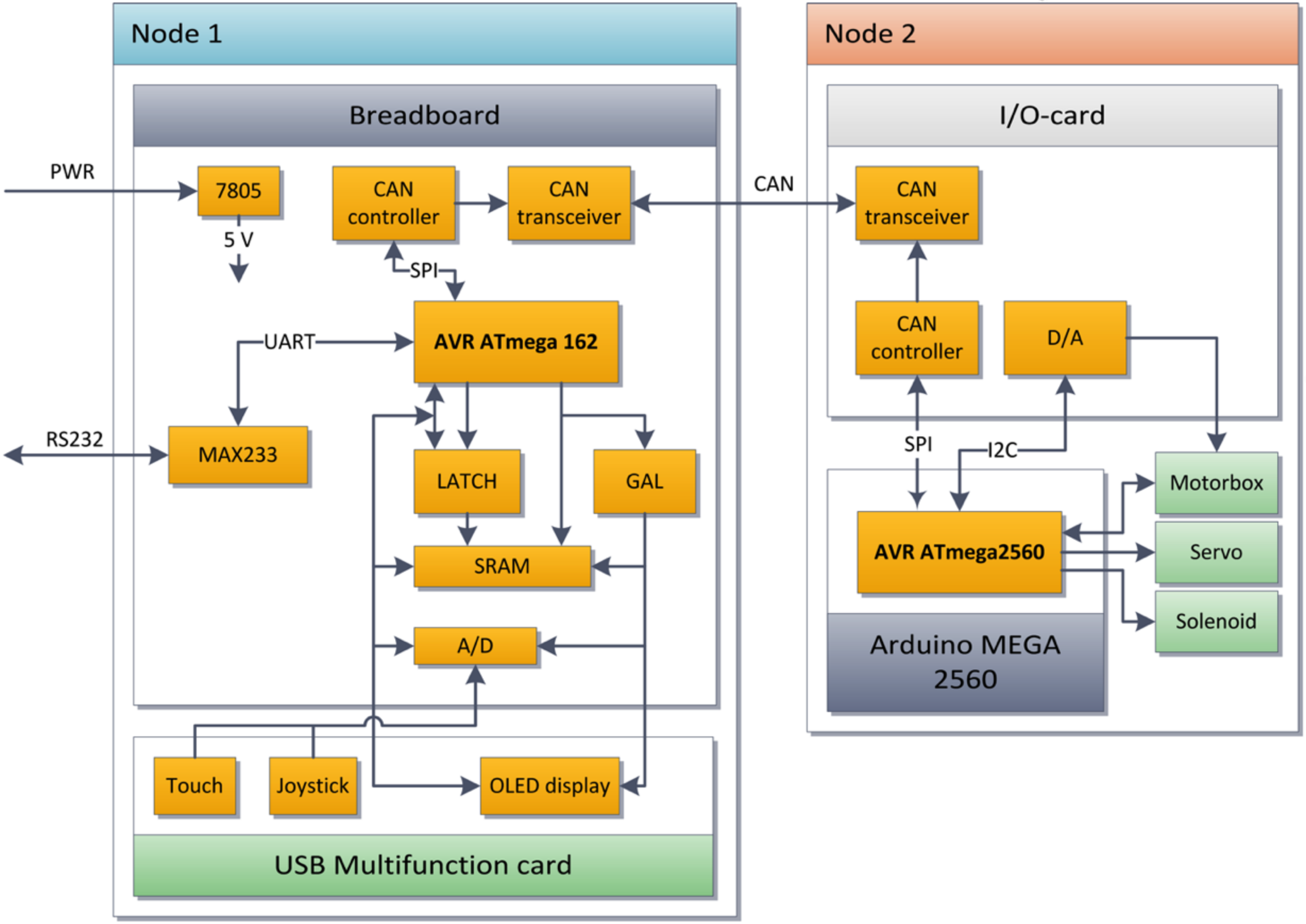
- Kolbjørn Austreng (B445??)

  Will act as proxy for Waseem when he is unavailable

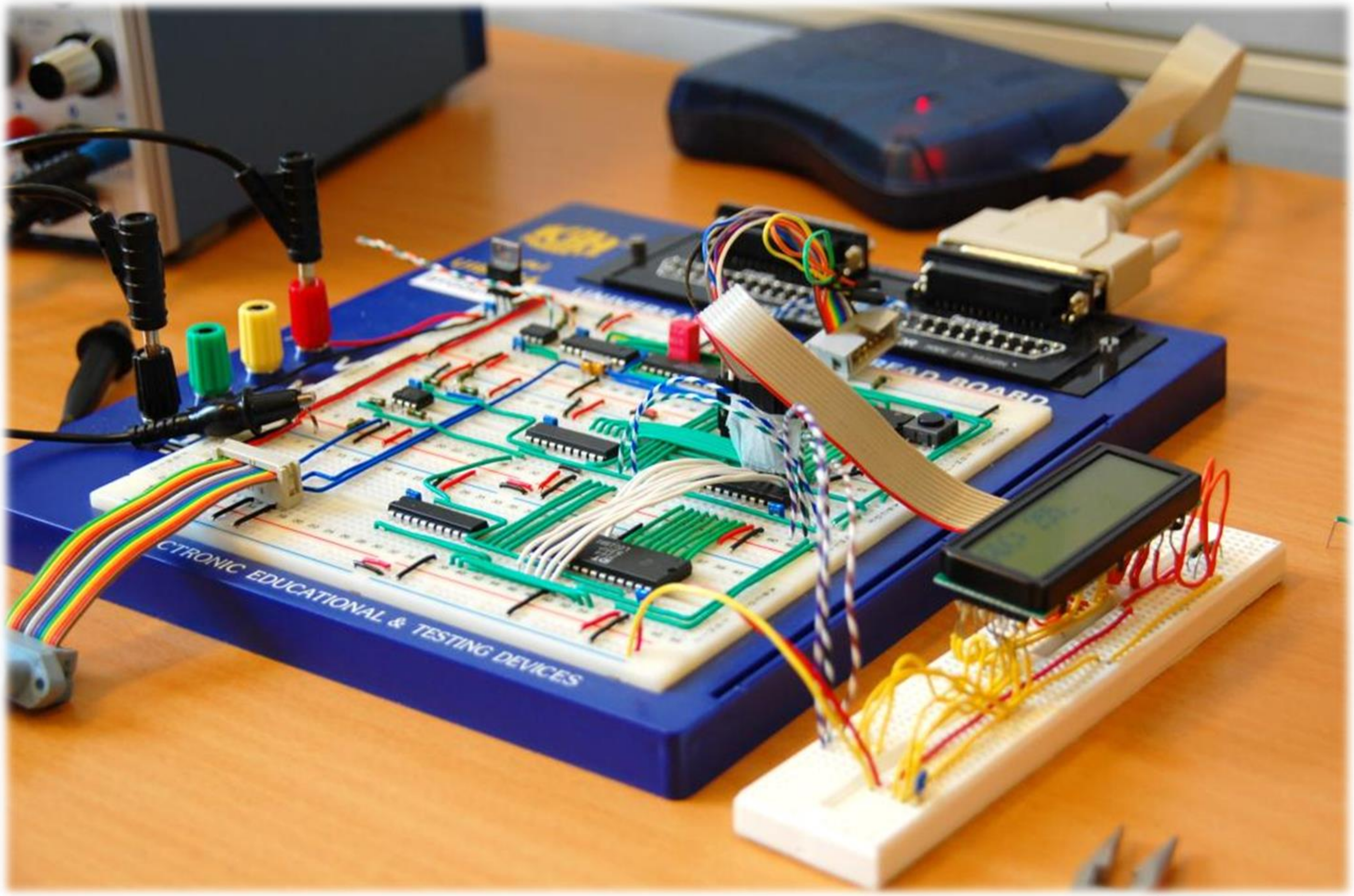
- SAs available 8 hours every lab day, minus a 1 hour break
  - Wed: 0900-1700; Thurs: 0900-1700; Fri: 0900-1700
  - The lab can be used whenever it is *free/available

Lab Project

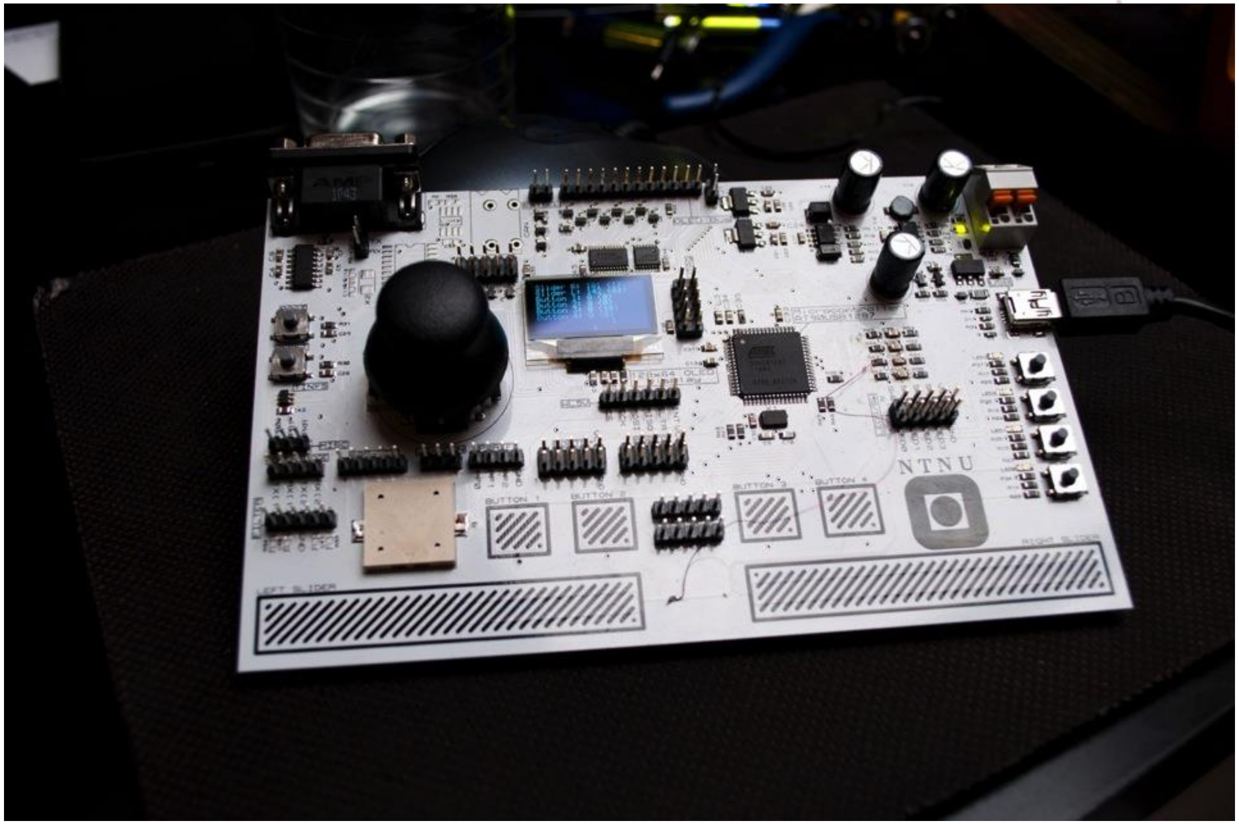
- A mechatronic project with the goal of building a computer controlled ping-pong game. Practical and fun!
- Tasks
  - Build two microcontroller based (embedded) nodes.
  - For one of the nodes, assemble discrete components on breadboard. ICs, resistors, capacitors etc.
  - For both of the nodes, develop software device drivers in C.
- Project counts **50%** of the final grade.


Node 1
Breadboard
PWR
7805
5 V
CAN controller
CAN transceiver
SPI
UART
AVR ATmega 162
RS232
MAX233
LATCH
GAL
SRAM
A/D
Touch
Joystick
OLED display
USB Multifunction card
CAN
Node 2
I/O-card
CAN transceiver
CAN controller
D/A
SPI
I2C
AVR ATmega2560
Arduino MEGA 2560
Motorbox
Servo
Solenoid

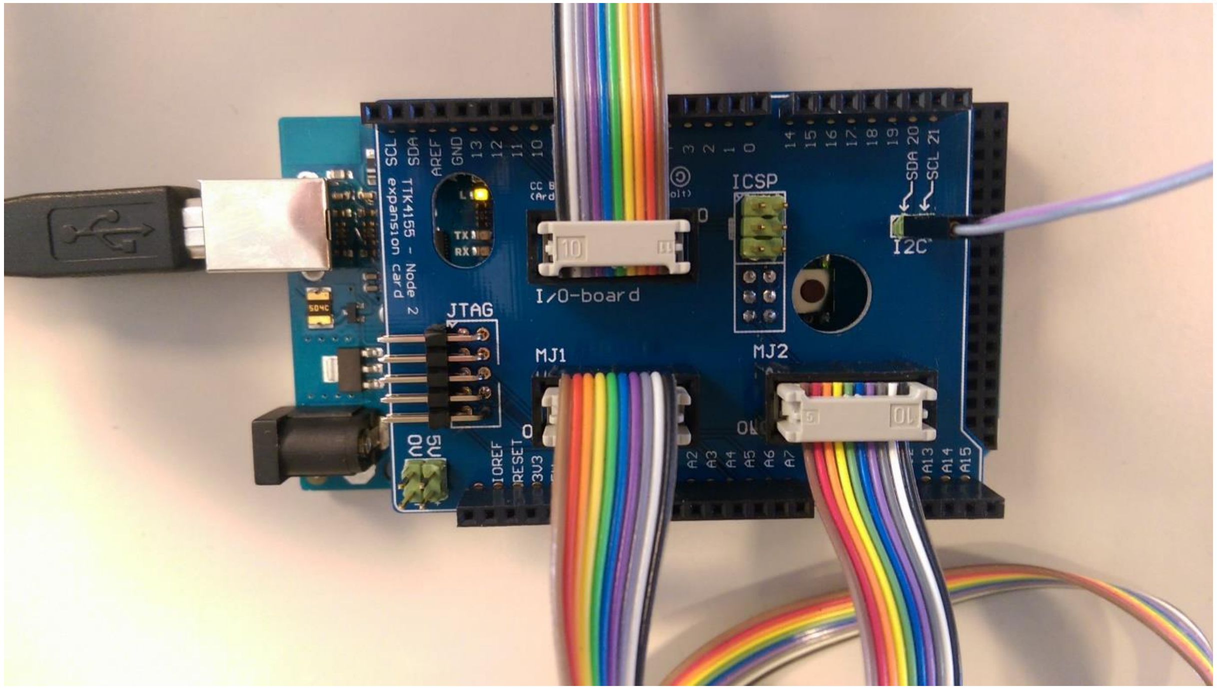
Node 1



The «USB Multifunction Board»



Node 2
The Arduino Mega 2560



Lab & component kit

- Lab: Elektro-building, G203 and G204 (up the stairs near “Infohjørnet” helpdesk) “Sanntidssalen”
  - No access?? send me an e-mail with your full name.
  - Reserved Wednesdays, Thursdays and Fridays.

- Kit: Handed out in room Elektro-D040 (basement)
  - Åsmund Stavdahl
  - Go there on your (first) lab day.
  - Deposit fee of 200 NOK, bring exact amount.
  - Must be returned in good condition.

Additional components

- Cables, headers, connectors etc. not in the kit
  - Available in the lab at SA desk.

- If you need more specific components, ask SAs they might help you

Remember to buy a padlock

- Lockable cabinets outside the lab
- Use the locker corresponding to your group number
- Don't lock in common lab equipment e.g. game boards

Lab assignments

- Eight assignments/exercises in total
  - Read lab manual thoroughly before starting each lab
  - Assignments/exercises are followed by a lab lecture => x4 lab lectures. Schedule on Blackboard

- Assignments must be approved by an SA
  - Try to get approved by the deadline
  - Helps you at the end e.g. if your project fails…

Evaluation

- All code submitted at the end of the project
  - Only use your own code except in the cases with permission

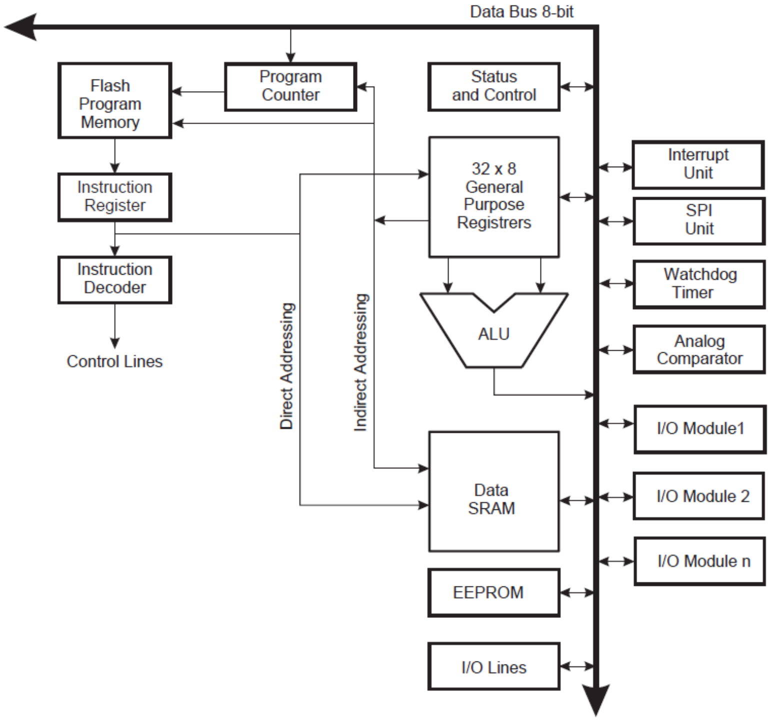
- No reports or written material required

- System demonstration and presentation (15 min)

- Score
  - 80% completion & functionality; 20 % extra features
  - Approved exercises does not guarantee a full score

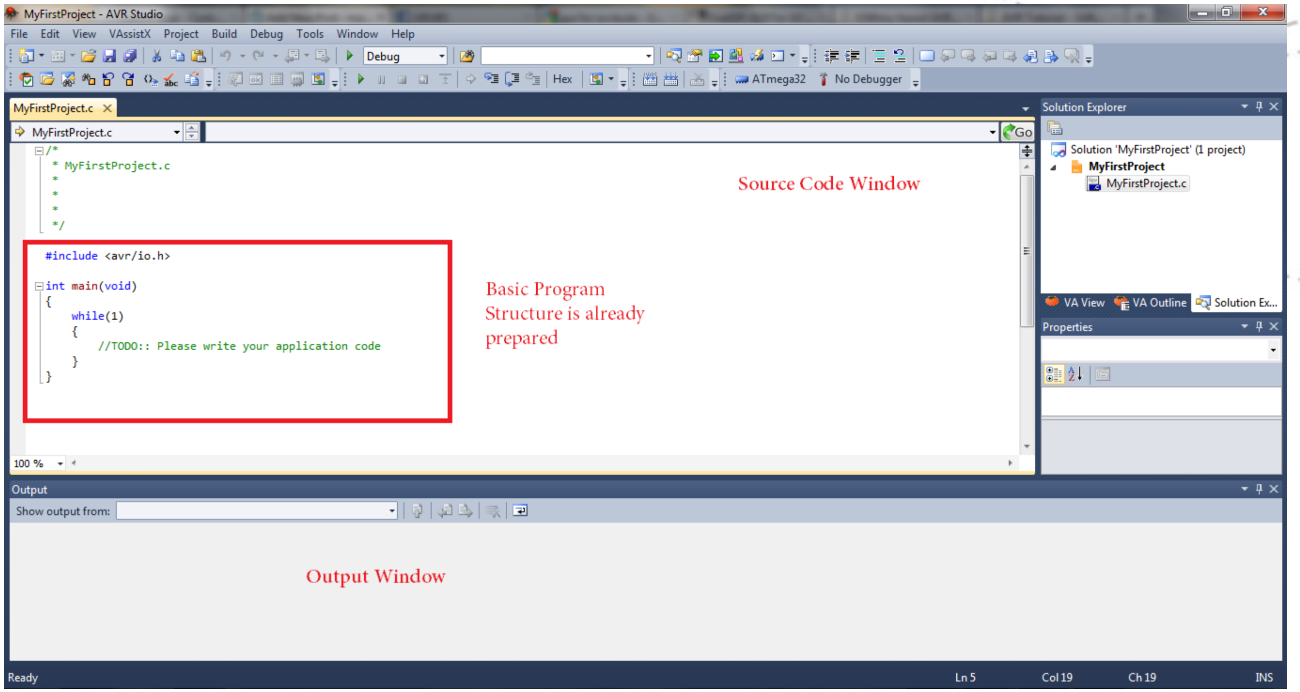
Microcontrollers

- A small computer on a single integrated circuit
  - Processor core
  - Memory
  - Programmable input/output units
    - Input/output pins we can control
    - Communication interfaces
    - ADCs
    - DACs
- Embedded applications
- This project: Atmel AVR family



Atmel Studio 6.2(Windows)

- Write code
- Compile
- (Simulate)
- Program
- *Debug



```
/*
 * MyFirstProject.c
 *
 *
 *
 */

#include <avr/io.h>

int main(void)
{
    while(1)
    {
        //TODO:: Please write your application code
    }
}
```

Source Code Window

Basic Program Structure is already prepared

Output Window

NTNU – Trondheim
Norwegian University of
Science and Technology

Atmel Linux toolchain

- SAs will also provide help for Linux based environment
- Follow this tutorial
- To facilitate compiling multiple source files and also programming, we will upload a make file on Bb

Exercise 1: Initial assembly of µC

- The microcontroller needs
  - Power => Voltage regulator.
  - Clock signal => Crystal oscillator.
  - Programming interface => JTAG (Atmel ICE).
- This will get the «main unit» up and running
- Test µC by using RS232/UART module
  - RS232 requires driver chip.
- Other units will then be connected in later assignments

UART & RS232

- Difference
  - UART => device used for asynchronous comm.
  - RS232 => standard, defines electrical aspects of comm. e.g. voltages, signaling, cables etc.
- RS232 standard interface, used quite often
- Makes your embedded system to communicate with other devices e.g. a PC etc.
- Useful in debugging. Make wrapper for printf function and you can print your desired status messages using standard C function

Recommended apporach

- Read lab manual before starting a lab exercise
- Use Atmel [datasheet]
- In case you are stuck, double check connections and basics from datasheet
  - Still need help?? ask SA
- Use QMS for getting help from an SA

UART Software Buffer

- Not required or mandatory just an advance method
- Read this tutorial and try to implement buffered UART. Again it is an extra in case you finish earlier and have time

UART s/w buffer

Reference Group

Any volunteers?? Preferably from different study programs

1) …
2) …
3) …

Lab Groups

- Group of THREE students with some groups having two students
- If you don’t have a group/members, talk with your neighbor student(s) and sign up as a group
- Will be published on Blackboard by tomorrow
- One person can signup/write names of all group members

Questions?

Auf wiedersehen...

Tips to work with the project

- ✓ Read the assignment before the each lab
- ✓ Master C and electronics
- ✓ Be careful and thorough when implementing each part
- ✓ Organize your code in compact drivers with logical interfaces
- ✓ Keep a nice and tidy breadboard
- ✓ Test frequently
- ✓ Use the web, but do not copy other people's work
- ✓ Read through the assignment text and perform the debugging steps before asking for help
- ✓ Read the datasheets carefully
- ✓ Go on with the next assignment when ready
- ✓ Make documentation!

Example: programming for AVR µCs in C

- I/O pins are organized as ports
- main()
- Bit manipulation
- Modularity
- Polling and interrupts

- Only the bit in position CS02 set

```
(1 << CS02)
```

- Only the bits in position COM01, COM00 and CS02 set

```
(1 << COM01) | (1 << COM00) | (0 << WGM01) | (1 << CS02)
```

- Set the bits CS02 and COM01 in TCCR0 register, clear the other bits

```
TCCR0 = (1 << CS02) | (1 << COM01);
```

- Set the bits CS02 and COM01, leave the other bits unchanged

```
TCCR0 |= (1 << CS02) | (1 << COM01);
```

- Clear the bit CS02, leave other bits unchanged

```
TCCR0 &= ~(1 << CS02);
```

HW Debugging Tips

- Power off the circuit before debugging.
- Check power and ground connections and also loose connections first => use DMM.
- Oscilloscope is a powerful tool. Use for time varying signals.
- Double check programmer’s connection. (Atmel ICE)
- Verify crossing of cables e.g. serial cable Tx and Rx.
- Read datasheets carefully and try to make the circuit as shown in the datasheets of IC(s).
- Tidy wiring helps in debugging.
  Do it from start.

NTNU – Trondheim
Norwegian University of
Science and Technology