Today:

- Binary numbers calculation
- Bitwise Operators
- Input – output ports
- Introduction to microprocessor programing in C

- You are expected to do parts 1, 2 and 3 described in the manual in the first 2 weeks of the labs and submit your notes for marking at the beginning of third week’s lab.
- If you finish early go on! The next parts are progressively more difficult. If you finish all the experiments early you will have more time for a great project!

Binary numbers

- Decimal numbers
- There are 10 digits 0-9
- Each digit represents consecutive power of 10
- $256 = 2x10^2 + 5x10^1 + 6x10^0$
- Binary numbers
- There are only 2 digits 0 and 1
- Each digit represents consecutive power of 2
- Binary $101 = 1x2^2+0x2^1+1x2^0 = 5$ in decimal

Operators

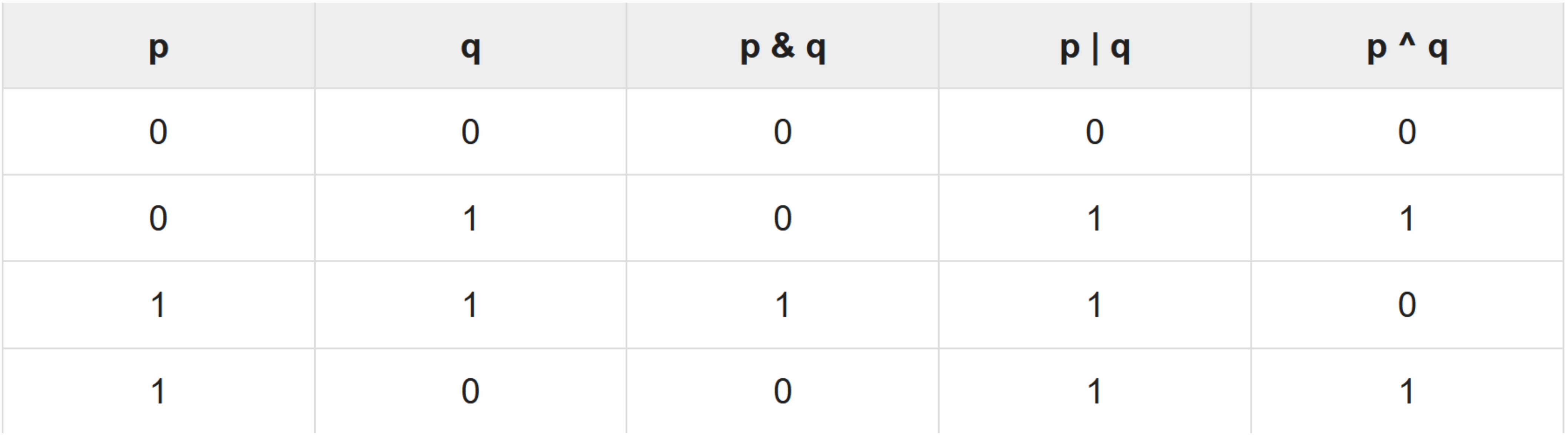
Arithmetic Operators:

=, +, -, *, /
% - modulus
5 mod 2 = 1
Reminder of dividing 5 by 2

Bitwise operators

& - bitwise AND
| - bitwise OR
^ - bitwise XOR (exclusive OR)
~ - bitwise NOT
<< - bitshift left
>> - bitshift right

Bitwise AND, OR and Exclusive OR



p	q	p & q	p | q	p ^ q
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

Comparison and Logical operators

Comparison:

==, <, >, !=, >=,<=

For example	if (i < 3), if (i != 3)
&& - logical AND	if (i == 1 && j == 2)
|| - logical OR	if (i == 1 || j == 2)
! logical NOT	

Input – output port

- Connection of 8 pins which can receive or output voltages.
- When we use it as digital port the only possible inputs or outputs are about 0 V or about 3.3 Volts.
- The way each pin works depends on the state of number of 8 bits in various memory locations.
- To set the state of each of these locations we send to it an 8 digit binary number.
- The binary number is indicated with prefix 0b

General Purpose Input/Output (GPIO) Ports

Several registers control the configuration and operation of sets of pins.
In these registers, the different bits in the register control different pins.
For our microprocessor x can be any integer between 1 and 8 as we have 8 GPIOs. Not all the pins are accessible.

PxDIR – sets the pin directions. Bit = 0 = input, Bit = 1 = output.

PxIN – input register. When configured for input, this register contains the digital input values

PxOUT – output register. When configured for output, writing to this register sets the outputs. When configured as input sets the pullup (1) or down (0)
PxREN – pullup/pulldown enable. Bit = 1, enable resistor (P1OUT sets whether pullup or down).
PxSEL – alternate function enable –0 means GPIO.
PxIE – Enable interrupt on some input port pins
PxIES – chooses if interrupt occurs on raising (0) or falling (1) edge

eg setting P1DIR = 0b00000011 configures pins P1.0 and P1.1 as outputs, P1.2-P1.7 as inputs

Programming ports in C

To decide which pin is output or input we set it to 1 or 0. It means sending a binary number to the port register PxDIR. To set the student number you have to add the commands, which will send the high (3.3V) or low (0V) to the pins corresponding to address and value of the digits corresponding to your student number using PxOUT = xxx commands after PxDIR commands set up the appropriate ports for output. Notice that the connection table indicates that we use 3 ports. With these series of commands you need to repeat what you did manually.

You will have to set the data and the address and while keeping them constant bring the strobe down and up again. How many commands does it require per digit?

Programming ports in C: Main

```
#include <msp430.h>             //this file provides the code with all the relevant names
                                //of the registers and their corresponding codes
void main(void) {               //start of the main program
WDTCTL = WDTPW + WDTHOLD;       // stop the watchdog timer (the watchdog timer
                                // is a feature that allows the cpu to detect and
                                // recover from some kinds of software bugs.
                                //We just want to disable it for now. )
  P1DIR = 0b00111111;     // All pins of port 1 are set to be outputs except P1.6 and 1.7

P1OUT = 0b00010001;     // Pins P1.0 and P1.4 will have 3.3.V
                        //pins 1.1, 1.2, 1.3 and 1.5 will have 0. Status of P1.6 and
                        // P1.7 will depend on what we connect there!
              }
```

Every C program must have a routine called main. The compiler generates the code necessary for the address of the main routine to go into the reset vector.

Programming ports in C: void

```
#include <msp430.h>              //this file provides the code with all the relevant names
                                 //of the registers and their corresponding codes
void main(void){                          //start of the main program
  WDTCTL = WDTPW + WDTHOLD;       // stop the watchdog timer (the watchdog timer
                                  // is a feature that allows the cpu to detect and
                                  // recover from some kinds of software bugs.
                                   //We just want to disable it for now. )

P1DIR =  0b11110111;       // All pins of port 1 are set to output except P1.3
P1OUT = 0b00010001;     // Pins P1.0 and P1.4 will have 3.3.V
                        //other pins will have 0. Status of P1.3 will depend on
                        //what we connect there!
                 }
```

The word "(void)" indicates that no parameters are passed to this function,
void here means that no values are returned by this function

Programming ports in C: brackets

```
#include <msp430.h>              //this file provides the code with all the relevant names
                                 //of the registers and their corresponding codes
void main(void) {                        //start of the main program
WDTCTL = WDTPW + WDTHOLD;        // stop the watchdog timer (the watchdog timer
                                 // is a feature that allows the cpu to detect and
                                 // recover from some kinds of software bugs.
                                 //We just want to disable it for now. )

P1DIR = 0b11110111;    // All pins of port 1 are set to output except P1.3
P1OUT = 0b00010001; // Pins P1.0 and P1.4 will have 3.3.V
                          //other pins will have 0. Status of P1.3 will depend on
                          //what we connect there!
      }
```

Brackets indicate the code, which belongs to main

Programming ports in C: semicolons

```
#include <msp430.h>              //this file provides the code with all the relevant names
                                 //of the registers and their corresponding codes
void main(void) {                         //start of the main program

WDTCTL = WDTPW + WDTHOLD;        // stop the watchdog timer (the watchdog timer
                                 // is a feature that allows the cpu to detect and
                                 // recover from some kinds of software bugs.
                                  //We just want to disable it for now. )


P1DIR = 0b11110111;          // All pins of port 1 are set to output except P1.3

P1OUT = 0b00010001;          // Pins P1.0 and P1.4 will have 3.3.V
                              //other pins will have 0. Status of P1.3 will depend on
                              //what we connect there!
                  }
```

Each C instruction should end with semicolon

Programming ports in C: comments

```
#include <msp430.h>              //this file provides the code with all the relevant names
                                 //of the registers and their corresponding codes
 void main(void) {                          //start of the main program

 WDTCTL = WDTPW + WDTHOLD;        // stop the watchdog timer (the watchdog timer
                                  // is a feature that allows the cpu to detect and
                                  // recover from some kinds of software bugs.
                                   //We just want to disable it for now. )


 P1DIR = 0b11110111;           // All pins of port 1 are set to output except P1.3

P1OUT = 0b00010001;            // Pins P1.0 and P1.4 will have 3.3.V
                                //other pins will have 0. Status of P1.3 will depend on
                                //what we connect there!
                  }
```

Programming ports in C: spaces and new lines

```
#include <msp430.h>              //this file provides the code with all the relevant names

                                  //of the registers and their corresponding codes
 void main(void) {                        //start of the main program

 WDTCTL = WDTPW + WDTHOLD;         // stop the watchdog timer (the watchdog timer
                                   // is a feature that allows the cpu to detect and
                                   // recover from some kinds of software bugs.
                                    //We just want to disable it for now. )



 P1DIR = 0b11110111;          // All pins of port 1 are set to output except P1.3

 P1OUT = 0b00010001;          // Pins P1.0 and P1.4 will have 3.3.V
                               //other pins will have 0. Status of P1.3 will depend on
                               //what we connect there!
                 }
```

Bitwise operators and calculator

https://miniwebtool.com/bitwise-calculator/

Notice that binary numbers do not have to be of equal length.

You have to use the same number of bits for calculation so if for example you need AND these two numbers: 10011001 AND 1110 you have to realize that1110 is the same as 00001110 (the same way as decimal 123 is the same as 0123).

10011001 AND 1110 = 00001000 because

10011001 AND 00001110 = 00001000

Activity 1due before lab of the second week of labs.

1. Which pins will be set to input and which to output after the command:

   P1DIR = 00110111

2. What is the result of bitwise XOR

   01000001 XOR 01

If and If else statments

```
integer n;

n=5;

if(n==5) n = n+1;  // the same as if(n==5) n += 1;

else n=0;


n=?
```

If and If else statments

```
integer n,m;
n=5;
if(n==4)   {
      n = n+1;
      m=7;}
else {
        n=0;
       m=0;
       }
n=?
m=?
```

For loop

```
int j;
for (j =16;j>0; j--) {
Some commands
                    }
```

These commands will be executed 16 times

```
if ( i == 0 ){
        a = 2;
        b = 3;
}
```

braces

As compared to:

```
if ( i  == 0 )
        a = 2;
        b = 3;
```

```
if (i == 0) a = 2;
b = 3;
```

Tabbing is helpful for readability.
Many editors have automatic tabbing

The compiler itself ignores whitespace – it's just for readability

```
if ( i == 0 ){
        a = 2;
        b = 3;
}
```

braces

vs

```
if ( i  == 0 )
        a = 2;
        b = 3;
```

executed even if
i != 0

```
if (i == 0) a = 2;
b = 3;
```

Tabbing is helpful for readability.
Many editors have automatic tabbing

The compiler itself ignores whitespace – it's just for readability

Example program Blink: while

```
#include <msp430.h>
void main(void) {
WDTCTL = WDTPW + WDTHOLD; //Stop WDT
P1DIR =0b00000001; //Set P1.0 to output;
P4DIR =0b10000000; //Set P4.7 to output;
P1OUT = 0b00000000; //Set the output Pin P1.0 to low
P4OUT = 0b10000000; //Set the output Pin P4.7 to high
while (1) { // Loop forever
_delay_cycles (500000); //This function introduces 0.5 s
delay
P1OUT = P1OUT ^ 0b00000001; //bitwise xor the output with
00000001   // can be also P1OUT^= 0b00000001;
P4OUT = P4OUT ^ 0b10000000; //bitwise xor the output with
10000000
}
}
```

Example program Blink: intrinsic

```
#include <msp430.h>
void main(void) {
WDTCTL = WDTPW + WDTHOLD; //Stop WDT
P1DIR =0b00000001; //Set P1.0 to output;
P4DIR =0b10000000; //Set P4.7 to output;
P1OUT = 0b00000000; //Set the output Pin P1.0 to low
P4OUT = 0b10000000; //Set the output Pin P4.7 to high
while (1) { // Loop forever
_delay_cycles (500000); //This function introduces 0.5 s delay
P1OUT = P1OUT ^ 0b00000001; //bitwise xor the output with 00000001
P4OUT = P4OUT ^ 0b10000000; //bitwise xor the output with 10000000
}
}
```