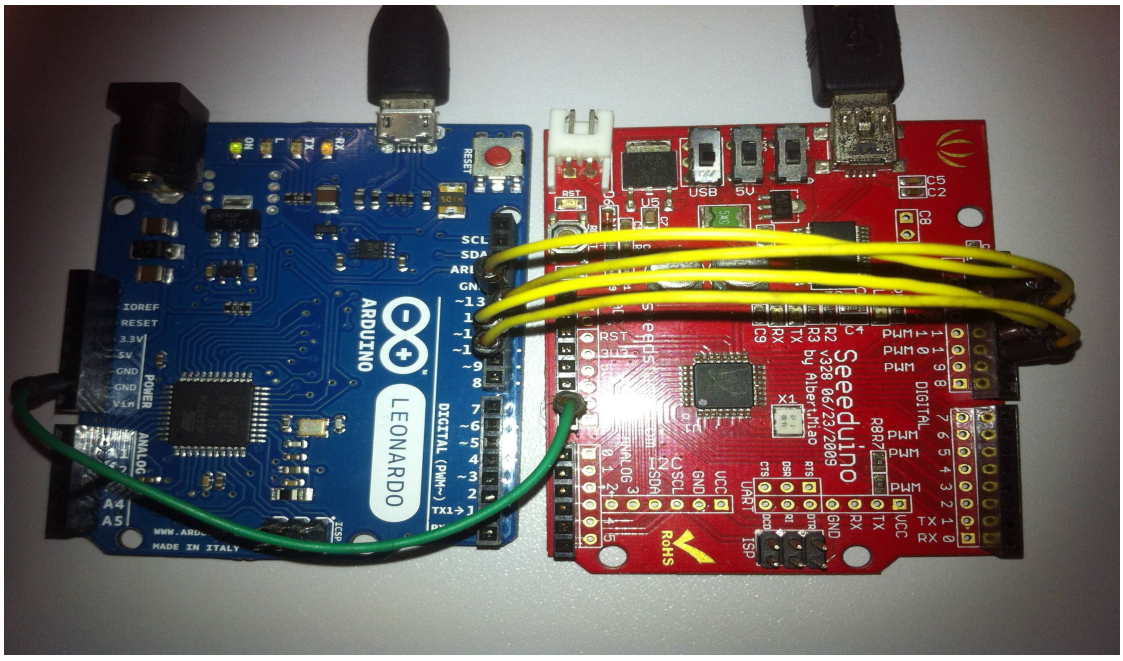
ΠΑΝ ΔΥΤ. ΜΑΚΕΔΟΝΙΑΣ
ΧΕΙΜ. ΕΞΑΜ. 2014-2015

ΤΜΗΜΑ ΜΗΧ. ΠΛΗΡΟΦΟΡΙΚΗΣ κ ΤΑΠ.
ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ
Διδάσκων: Κοντογιάννης Σωτήριος

ΕΡΓΑΣΤΗΡΙΟ 8

ΑΣΚΗΣΗ 1: Να δημιουργήσετε ένα πρόγραμμα σε C το οποίο θα χρησιμοποιεί την βιβλιοθήκη SPI ώστε να μεταφέρει μέσω της SPI θύρας εισόδου εξόδου δύο arduino uno (master – slave) Το μήνυμα "Hello world!".

Συνδεσμολογία:



Υπόδειξη 1 spi_server:

```
#include <SPI.h>

void setup (void)
{
  Serial.begin(115200);
  digitalWrite(SS, HIGH);
  SPI.begin ();
  SPI.setClockDivider(SPI_CLOCK_DIV8);
}

void loop (void)
{
  char c;

  // enable Slave Select
  digitalWrite(SS, LOW);    // SS is pin 10

  // send test string
  for (const char * p = "Hello, world!\n" ; c = *p; p++) {
    SPI.transfer (c);
    Serial.print(c);
  }

  // disable Slave Select
  digitalWrite(SS, HIGH);

  delay (1000);
}
```

Υπόδειξη 1 spi_client:

```
#include <SPI.h>

char buf [100];
volatile byte pos;
volatile boolean process_it;

void setup (void)
{
  Serial.begin (115200);

  // have to send on master in, *slave out*
  pinMode(MISO, OUTPUT);

  // turn on SPI in slave mode
  SPCR |= _BV(SPE);

  // get ready for an interrupt
  pos = 0;   // buffer empty
  process_it = false;

  // now turn on interrupts
  SPI.attachInterrupt();
}

// SPI interrupt routine
ISR (SPI_STC_vect)
{
  byte c = SPDR;  // grab byte from SPI Data Register
  if (pos < sizeof buf)
  {
    buf [pos++] = c;
    if (c == '\n')
      process_it = true;
  }
}

void loop (void)
{
  if (process_it)
  {
    buf [pos] = 0;
    Serial.println (buf);
    pos = 0;
    process_it = false;
  }
}
```

ΑΣΚΗΣΗ 2: Να δημιουργήσετε ένα πρόγραμμα σε C/assembly το οποίο θα χρησιμοποιεί την βιβλιοθήκη SPI ώστε να μεταφέρει μέσω της SPI θύρας εισόδου εξόδου δύο arduino uno (master – slave) Τον αριθμό 0x01.

SERVER:

```
#include <avr/io.h>
#include <util/delay.h>
#define SPI_PORT PORTB
#define SPI_DDR  DDRB
#define SPI_CS   PINB2

unsigned char SPI_WriteRead(unsigned char dataout)
{
  unsigned char datain;
  // Start transmission (MOSI)
  SPDR = dataout;
  // Wait for transmission complete
  while(!(SPSR & (1<<SPIF)));
  // Get return Value;
  datain = SPDR;
  // Latch the Output using rising pulse to the RCK Pin
  SPI_PORT |= (1<<SPI_CS);
    _delay_us(1);              // Hold pulse for 1 micro second
  // Disable Latch
  SPI_PORT &= ~(1<<SPI_CS);
   // Return Serial In Value (MISO)
  return datain;
}
```

```
void setup()
{
  Serial.begin(9600);
  // Set MOSI and SCK as output, others as input
  SPI_DDR = (1<<DDB3)|(1<<DDB5)|(1<<DDB2);
  // Latch Disable (RCK Low)
  SPI_PORT &= ~(1<<SPI_CS);
  // Enable SPI, Master, set clock rate fck/2 (maximum)
  SPCR = (1<<SPE)|(1<<MSTR)|(1<DORD);
  SPSR = (1<<SPI2X);

}

void loop()
{
  // Latch Disable (RCK Low)
  SPI_PORT &= ~(1<<SPI_CS);
   Serial.println("Starting Transmission:");
  SPI_WriteRead('0');
  _delay_ms(1000);
}
```

CLIENT:

```
#include <avr/io.h>
#include <util/delay.h>
#define SPI_PORT PORTB
#define SPI_DDR  DDRB
#define SPI_CS   PINB2
unsigned char cnt;

unsigned char SPI_WriteRead(unsigned char dataout)
{
  unsigned char datain;
  // Start transmission (MOSI)
  SPDR = dataout;
  // Wait for transmission complete
  while(!(SPSR & (1<<SPIF)));
  // Get return Value;
  datain = SPDR;
  // Latch the Output using rising pulse to the RCK Pin
  SPI_PORT |= (1<<SPI_CS);
    _delay_us(1);              // Hold pulse for 1 micro second
  // Disable Latch
  SPI_PORT &= ~(1<<SPI_CS);
   // Return Serial In Value (MISO)
  return datain;
}
```

```
void setup()
{
  Serial.begin(9600);
  /* Set MISO output, all others input */
  SPI_DDR |= (1<<DDB4);
  /* Enable SPI, the ATmega is in Slave mode at startUp */
  SPCR |= (1<<SPE);
  Serial.println("Receiving Transmission:");
}

void loop()
{
  cnt=SPI_WriteRead(0);
  Serial.println(cnt);
  _delay_ms(500);
}
```

ASSEMBLY SERVER:

spi3.h

```
/*
 * Global register variables.
 */
#ifdef __ASSEMBLER__

/* Assembler-only stuff */

#else  /* !ASSEMBLER */

/* C-only stuff */

#include <stdint.h>
#include <WString.h> // for String type
extern "C" void spi_send(void);
extern "C" void spi_init(void);
#endif /* ASSEMBLER */
```

spi3.ino

```
#include "spi3.h"

void setup()
{
  Serial.begin(9600);
   while (!Serial) {}
  spi_init();
   Serial.println("Server Start");
  delay(1000);

}
void loop()
{
  Serial.println("Sending data");
  spi_send();
  delay(2000);
}
```

code.S

```
#define __SFR_OFFSET 0
/*Automatically subtract 0x20 from I/O addresses
*/
#include <avr/io.h>
#include <avr/interrupt.h>
#include "spi3.h"

#define SPI_PORT PORTB
#define SPI_DDR  DDRB
#define SPI_CS   PINB2

.section .data
.section .text
.global spi_init
spi_init:
;******************MASTER SPI*****************
;*********************************************
; DECLARE MASTER SPI :
LDI R16,(1<<DDB2)|(1<<DDB3)|(1<<DDB5)
OUT DDRB,R16
LDI R17,(1<<SPE)|(1<<MSTR);|(1<<DORD)
OUT SPCR,R17
LDI R18,(1<<SPI2X)

OUT SPSR,R18
;*********************************************
ret
.global spi_send
spi_send:
send:cbi PORTB,SPI_CS
ldi r19,0x04
out SPDR,r19
nop
nop
nop
ldi r20,0x00
wait: inc r20
cpi r20,0xFF
breq mend
rjmp wait
mend:sbi PORTB,2
ret
.end
```