```c
#include <avr/io.h>
#include <avr/interrupt.h>
#include <stdint.h>
#include <avr/delay.h>

#define spiWREN 0x06
#define spiWRITE 0x02
#define spiREAD 0x03
#define spiWRDI 0x04

//////////////////////
///SPI függvények///
//////////////////////

void SPI_Init(void)
{
	/* Set MOSI and SCK output, all others input */
	DDRB = (1<<DDB5)|(1<<DDB7);
	/* Enable SPI, Master, set clock rate fck/4 */
	SPCR = (1<<SPE)|(1<<MSTR);  /*|(1<<SPR0)*/
}

void SPI_Transmit(char cData)
{
	/* Start transmission */
	SPDR = cData;
	/* Wait for transmission complete */
	while(!(SPSR & (1<<SPIF)));
	//while(SPIF == 0);
}

void SPI_Write(int adr, int dat)
{
	int dummy;
	cbi(PORTA,0); //eeprom engedélyezés
	SPI_Transmit(6); //WREN
	sbi(PORTA,0); //eeprom tiltás

	cbi(PORTA,0); //eeprom engedélyezés
	SPI_Transmit(2); //WRITE
	SPI_Transmit(0); // Cím 15...8
	SPI_Transmit(adr); // Cím 7...0
	SPI_Transmit(dat); // Adat
	sbi(PORTA,0); //eeprom tiltás
}

int SPI_Read(int adr)
{
	int dummy, dat;
	cbi(PORTA,0); //eeprom engedélyezés
	SPI_Transmit(3); //READ
	SPI_Transmit(0); // Cím 15...8
	SPI_Transmit(adr); // Cím 7...0
	SPI_Transmit(0); // Küldök valamit, hogy a Slave töltse az adatot az SPDR-be
	dat = SPDR;
	sbi(PORTA,0); //eeprom tiltás

	return(dat);
}
```

```
int main(void)
{
	int data, adr;

	PORTA = 0x00;
	DDRA = 0xff; //output
	sbi(PORTA,0); //CS at logic one

	SPI_Init();

	SPI_Write(3,7); //a 3. helyre írjon 7-et
	sbi(PORTA,1); //indikátor1
	data = SPI_Read(3); //a 3. helyről olvassa ki az adatot
	sbi(PORTA,2); //indikátor2
	UartTransmit(data);


	while(1)
	{

	}
}
```