Källkod

```
#include <avr/io.h>
#define F_CPU 1000000UL  // 1 MHz
#include <util/delay.h>

void set_pin(char port, char pin, char state);
unsigned short int a;
void read(unsigned short int addressing, int
reset)
{
   /* Read to Register */
   if( reset == 1) {
      set_pin('A', PA2, 1);
      set_pin('A', PA2, 0);
      set_pin('A', PA2, 1);
   }

   PORTB = addressing; // addressinG
      DDRD = 0b00000000;
   set_pin('A', PA5, 0);
   set_pin('A', PA5, 1);
   set_pin('A', PA1, 0);
}

void write(unsigned short int  addressing,
unsigned short int  databus, int reset)
{

   // Write to the DUART shizzzle

   if( reset == 1) {
      set_pin('A', PA2, 1);
      set_pin('A', PA2, 0);
      set_pin('A', PA2, 1);
   }

   set_pin('A', PA5, 1);
   set_pin('A', PA1, 1);

   PORTB = addressing; // addressinG
   PORTD = databus; // databus

   set_pin('A', PA5, 0);
   set_pin('A', PA1, 0);

   set_pin('A', PA1, 1);
   set_pin('A', PA5, 1);


}

void tx_uart(char val)
{

   set_pin('A', PA5, 1);
   set_pin('A', PA1, 1);

   PORTB = 0b11000000; // addressinG
   PORTD = val; // databus

   set_pin('A', PA5, 0);
   set_pin('A', PA1, 0);

   set_pin('A', PA1, 1);
   set_pin('A', PA5, 1);
}

unsigned short int clock_Crypto()
{
   unsigned short int slask;

     set_pin('B', PB2, 0);
   set_pin('B', PB2, 1);

   slask=PINB&0b00000010;
   if(slask != 0x00){
      return 0b00000001;
   } else {
      return 0b00000000;
   }
}

void tx_uart_encrypt()
{
   unsigned short int final;
   int dum = 0;
```

```
  unsigned short int array[] = {0b10000000,
0b01000000, 0b00100000, 0b00010000,
0b00001000, 0b00000100,
0b0000010,0b00000001};
  unsigned short int zero = 0b00000000;
//0b01000110

  for(int k = 0; k <= 7; k++) {
    a = clock_Crypto();
    if(a == 0b00000000) {
      final = final | zero;
      dum++;
      }
    else {
      final = final | array[k];
      dum--;
      }
  }


  set_pin('A', PA5, 1);
  set_pin('A', PA1, 1);

  PORTB = 0b11000000; // addressinG
  PORTD = final;

  set_pin('A', PA5, 0);
  set_pin('A', PA1, 0);

  set_pin('A', PA1, 1);
  set_pin('A', PA5, 1);
}

int rx_uart()
{
  char val;

  set_pin('A', PA1, 1);
  set_pin('A', PA5, 1);

  PORTB = 0b11000000; // addressinG
  DDRD = 0b00000000;

  set_pin('A', PA5, 0);
  set_pin('A', PA5, 1);
  set_pin('A', PA1, 0);

  val = PORTD;
  DDRD = 0b11111111;
  return val;
}

void init_Crypto()
{
  int key[] = {1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1,
0, 1, 0};
  int key1[] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
  set_pin('B', PB3, 0);
  set_pin('B', PB3, 1);

  for( int k = 0; k <= 15; k++ ) {
    if(key[k] == 1) {
      set_pin('B', PB0, 1);

    } else {
      set_pin('B', PB0, 0);
    }
    set_pin('B', PB2, 0);
    set_pin('B', PB2, 1);
  }
}


void init_duart()
{
  //Setting the Data Direction Registers
  DDRA = 0b11111111;
  DDRB = 0b11111101;
  DDRC = 0b00000111;
  DDRD = 0b11111111;

  // Write to MR1A
  write(0b00000000, 0b00010011, 1);
//0b00010010

  // Write to MR2A
  write(0b00000000, 0b00000111, 0);
//0b00010111
```

```
   // Write to CRA
   write(0b01000000, 0x0A, 0); //0b00000101

   write(0b01000000, 0x80, 0);

   // Write to ACR
   write(0b00100000, 0b10110000, 0);
//0b11000000

      // Write to CSRA
   write(0b10000000, 0xBB, 0); //

   // Write to CRA
   write(0b01000000, 0x05, 0); //0b00000101
}

void main(void)
{

   init_Crypto();
   set_pin('C', PC1, 1);
    init_duart();

   while(1)
   {
   set_pin('C', PC1, 0);
      clock_Crypto();
      //_delay_ms(10);
      tx_uart(0b01000110);
      //_delay_ms(10);
      tx_uart_encrypt();
      _delay_ms(10);
      //tx_uart('D');
      _delay_ms(10);
      //tx_uart('R');
      _delay_ms(10);
      //tx_uart('I');
      _delay_ms(10);
      //tx_uart('K');
      _delay_ms(1000);
      //tx_uart(' ');

      set_pin('C', PC1, 1);
   }
   return;
}

void set_pin(char port, char pin, char state){
   char set = 1 << pin;
   if(port == 'A'){
      set &= PORTA;
      if(set && !state){ //ändra från 1 -> 0
         PORTA ^= set;
      }
      if(set == 0 && state){ //ändra från 0 -> 1
         set = 1 << pin;
         PORTA ^= set;
      }
   }
   else if(port == 'B'){
      set &= PORTB;
      if(set && !state){ //ändra från 1 -> 0
         PORTB ^= set;
      }
      if(set == 0 && state){ //ändra från 0 -> 1
         set = 1 << pin;
         PORTB ^= set;
      }
   }
   else if(port == 'C'){
      set &= PORTC;
      if(set && !state){ //ändra från 1 -> 0
         PORTC ^= set;
      }
      if(set == 0 && state){ //ändra från 0 -> 1
         set = 1 << pin;
         PORTC ^= set;
      }
   }
   else if(port == 'D'){
      set &= PORTD;
      if(set && !state){ //ändra från 1 -> 0
         PORTD ^= set;
      }
      if(set == 0 && state){ //ändra från 0 -> 1
         set = 1 << pin;
         PORTD ^= set;
      }
   }
}
```