```
.INCLUDE "M8DEF.INC"
.CSEG
.ORG 0

LDI R16,LOW(RAMEND)
OUT SPL,R16

LDI R16,HIGH(RAMEND)
OUT SPH,R16

LDI R16,1
OUT DDRC,R16            ;TESTIGO PARA FLASH
OUT PORTC,R16

FIN: RJMP FIN
```

```
:100000000FE50DBF04E00EBF01E004BB05BBFFCF51
:10001000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF0
:10002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE0
:10003000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFD0
:10004000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC0
:10005000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFB0
:10006000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFA0
:10007000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF90
:10008000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF80
:10009000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF70
```

Contenido de la aplicación

```
:1000A000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF60
:1000B000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF50
:1000C000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF40
:1000D000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF30
:1000E000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF20
:1000F000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF10
:10010000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
:10011000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEF
:10012000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFDF
:10013000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFCF
                        .
                        .
                        .
```

```
;Programa de prueba para SMALLBOOTSTART
.INCLUDE "M8DEF.INC"
.CSEG                   ;OPCIONAL
.ORG SMALLBOOTSTART

LDI R16,LOW(RAMEND)
OUT SPL,R16

LDI R16,HIGH(RAMEND)
OUT SPH,R16
```

Esta sección la veremos reflejada en la sección del SMALLBOOT

```
LDI R16,1
OUT DDRB,R16          ;TEST PARA BOOTLOADER EN PB0
OUT PORTB,R16

FIN: RJMP FIN
```

```
:10000000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF00
:10001000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF0
:10002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE0
:10003000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFD0
:10004000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC0
:10005000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFB0
:10006000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFA0
:10007000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF90
:10008000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF80
:10009000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF70
:1000A000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF60
                    .
                    .
                    .
:101EF000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF2
:101F00000FE50DBF04E00EBF01E007BB08BBFFCF2C
:101F1000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFD1
:101F2000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC1
:101F3000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFB1
:101F4000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFA1
:101F5000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF91
:101F6000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF81
:101F7000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF71
:101F8000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF61
:101F9000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF51
:101FA000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF41
:101FB000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF31
:101FC000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF21
:101FD000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF11
:101FE000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF01
:101FF000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF1
:00000001FF
```

Contenido del Bootloader

```
.INCLUDE "M8DEF.INC"
.CSEG
.ORG 0                      ;DIRECCIÓN PARA LA APLICACIÓN

LDI R16,LOW(RAMEND)
OUT SPL,R16

LDI R16,HIGH(RAMEND)
OUT SPH,R16

LDI R16,1
OUT DDRC,R16                ;TEST PARA FLASH
OUT PORTC,R16

.ORG SMALLBOOTSTART         ;DIRECCIÓN PARA BOOTLOADER

LDI R16,1
OUT DDRB,R16                ;TEST PARA BOOTLOADER
OUT PORTB,R16

FIN: RJMP FIN
```

```
:100000000FE50DBF04E00EBF01E004BB05BBFFFF21
:10001000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF0
:10002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE0
:10003000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFD0
:10004000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC0
:10005000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFB0
:10006000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFA0
:10007000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF90
:10008000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF80
:10009000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF70
:1000A000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF60
:1000B000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF50
:1000C000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF40
                .
                .
                .
```

Contenido de la aplicación

```
:101EF000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF2
:101F000001E007BB08BBFFCFFFFFFFFFFFFFFFA5
:101F1000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFD1
:101F2000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC1
:101F3000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFB1
:101F4000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFA1
:101F5000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF91
:101F6000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF81
:101F7000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF71
:101F8000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF61
:101F9000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF51
:101FA000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF41
:101FB000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF31
:101FC000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF21
:101FD000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF11
:101FE000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF01
:101FF000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF1
:00000001FF
```

Contenido del Bootloader

**Archivo ATmegaBOOT.C (reducido):**

```
/*************************************************************/
/* Serial Bootloader for Atmel mega8 AVR Controller          */
/*                                                           */
/* ATmegaBOOT.c                                              */
/*                                                           */
/* Copyright (c) 2003, Jason P. Kyle                         */
/*                                                           */
/* Hacked by DojoCorp - ZGZ - MMX - IVR                      */
/* Hacked by David A. Mellis                                 */
/*                                                           */
/* This program is free software; you can redistribute it */
/* and/or modify it under the terms of the GNU General    */
/* Public License as published by the Free Software       */
/* Foundation; either version 2 of the License, or        */
/* (at your option) any later version.                    */
/*                                                           */
/* This program is distributed in the hope that it will   */
/* be useful, but WITHOUT ANY WARRANTY; without even the  */
/* implied warranty of MERCHANTABILITY or FITNESS FOR A   */
/* PARTICULAR PURPOSE.  See the GNU General Public        */
/* License for more details.                              */
/*                                                           */
/* You should have received a copy of the GNU General     */
/* Public License along with this program; if not, write  */
/* to the Free Software Foundation, Inc.,                 */
/* 59 Temple Place, Suite 330, Boston, MA  02111-1307 USA */
/*                                                           */
/* Licence can be viewed at                               */
/* http://www.fsf.org/licenses/gpl.txt                    */
/*                                                           */
/* Target = Atmel AVR m8                                  */
/*************************************************************/

#include <inttypes.h>
#include <avr/io.h>
#include <avr/io.h>
#include <avr/pgmspace.h>
#include <avr/eeprom.h>
#include <avr/interrupt.h>
#include <util/delay.h>

//#define F_CPU                16000000
```

```
/* We, Malmoitians, like slow interaction
 * therefore the slow baud rate ;-)
 */
//#define BAUD_RATE            9600

/* 6.000.000 is more or less 8 seconds at the
 * speed configured here
 */
//#define MAX_TIME_COUNT 6000000
#define MAX_TIME_COUNT (F_CPU>>1)
///#define MAX_TIME_COUNT_MORATORY   1600000

/* SW_MAJOR and MINOR needs to be updated from time to time
to avoid warning message from AVR Studio */
#define HW_VER      0x02
#define SW_MAJOR 0x01
#define SW_MINOR 0x12

// AVR-GCC compiler compatibility
// avr-gcc compiler v3.1.x and older doesn't support outb()
and inb()
//      if necessary, convert outb and inb to outp and inp
#ifndef outb
      #define outb(sfr,val)  (_SFR_BYTE(sfr) = (val))
#endif
#ifndef inb
      #define inb(sfr) _SFR_BYTE(sfr)
#endif

/* defines for future compatibility */
#ifndef cbi
      #define cbi(sfr, bit) (_SFR_BYTE(sfr) &= ~_BV(bit))
#endif
#ifndef sbi
      #define sbi(sfr, bit) (_SFR_BYTE(sfr) |= _BV(bit))
#endif

/* Adjust to suit whatever pin your hardware uses to enter
the bootloader */
#define eeprom_rb(addr)   eeprom_read_byte ((uint8_t *)(addr))
#define eeprom_rw(addr)   eeprom_read_word ((uint16_t *)
(addr))
#define eeprom_wb(addr, val)   eeprom_write_byte ((uint8_t *)
(addr), (uint8_t)(val))
```

```
/* Onboard LED is connected to pin PB5 */
#define LED_DDR  DDRB
#define LED_PORT PORTB
#define LED_PIN  PINB
#define LED      PINB5


#define SIG1 0x1E  // Yep, Atmel is the only manufacturer of
AVR micros.  Single source :(
#define SIG2 0x93
#define SIG3 0x07
#define PAGE_SIZE  0x20U //32 words


void putch(char);
char getch(void);
void getNch(uint8_t);
void byte_response(uint8_t);
void nothing_response(void);

union address_union {
  uint16_t word;
  uint8_t  byte[2];
} address;

union length_union {
  uint16_t word;
  uint8_t  byte[2];
} length;

struct flags_struct {
  unsigned eeprom : 1;
  unsigned rampz  : 1;
} flags;

uint8_t buff[256];
//uint8_t address_high;

uint8_t pagesz=0x80;

uint8_t i;
//uint8_t bootuart0=0,bootuart1=0;


void (*app_start)(void) = 0x0000;
```

```
int main(void)
{
  uint8_t ch,ch2;
  uint16_t w;

  //cbi(BL_DDR,BL);
  //sbi(BL_PORT,BL);

  asm volatile("nop\n\t");

  /* check if flash is programmed already, if not start boot-
loader anyway */
  //if(pgm_read_byte_near(0x0000) != 0xFF) {

    /* check if bootloader pin is set low */
    //if(bit_is_set(BL_PIN,BL)) app_start();
  //}

  /* initialize UART(s) depending on CPU defined */
  /* m8 */
  UBRRH = (((F_CPU/BAUD_RATE)/16)-1)>>8;  // set baud rate
  UBRRL = (((F_CPU/BAUD_RATE)/16)-1);
  UCSRB = (1<<RXEN)|(1<<TXEN);  // enable Rx & Tx
  UCSRC = (1<<URSEL)|(1<<UCSZ1)|(1<<UCSZ0);  // config USART;
8N1

  //UBRRL = (uint8_t)(F_CPU/(BAUD_RATE*16L)-1);
  //UBRRH = (F_CPU/(BAUD_RATE*16L)-1) >> 8;
  //UCSRA = 0x00;
  //UCSRC = 0x86;
  //UCSRB = _BV(TXEN)|_BV(RXEN);

.
.
.

            /* Universal SPI programming command, disabled.
Would be used for fuses and lock bits.  */
            else if(ch=='V') {
              getNch(4);
              byte_response(0x00);
            }

            /* Write memory, length is big endian and is in
bytes  */
```

```
            else if(ch=='d') {
              length.byte[1] = getch();
              length.byte[0] = getch();
              flags.eeprom = 0;
              if (getch() == 'E') flags.eeprom = 1;
              for (w=0;w<length.word;w++) {
              buff[w] = getch();// Store data in buffer,
can't keep up with serial //data stream whilst programming
pages
              }
              if (getch() == ' ') {
                        if (flags.eeprom) {  //Write to EEPROM
one byte at a time
                              for(w=0;w<length.word;w++) {
                                    eeprom_wb(address.word,-
buff[w]);
                                    address.word++;
                              }
                        } else {      //Write to FLASH one
page at a time
                              //if (address.byte[1]>127) ad-
dress_high = 0x01;      //Only possible with m128, m256 will
need 3rd address byte. FIXME
                              //else address_high = 0x00;

                              //address.word = address.word
<< 1;
                              //address * 2 -> byte location
                              //if ((length.byte[0] & 0x01))
length.word++;.
.
.
.

void putch(char ch)
{
  /* m8 */
  while (!(inb(UCSRA) & _BV(UDRE)));
  outb(UDR,ch);
}

char getch(void)
{
  /* m8 */
      uint32_t count = 0;
```

```
  while(!(inb(UCSRA) & _BV(RXC))) {
            /* HACKME:: here is a good place to count times*/
            count++;
            if (count > MAX_TIME_COUNT)
                  app_start();
  }
  return (inb(UDR));
}

void getNch(uint8_t count)
{
  uint8_t i;
  for(i=0;i<count;i++) {
    /* m8 */
    //while(!(inb(UCSRA) & _BV(RXC)));
    //inb(UDR);
            getch(); // need to handle time out
  }
}

void byte_response(uint8_t val)
{
  if (getch() == ' ') {
    putch(0x14);
    putch(val);
    putch(0x10);
  }
}

void nothing_response(void)
{
  if (getch() == ' ') {
    putch(0x14);
    putch(0x10);
  }
}

/* end of file ATmegaBOOT.c */
```

**Archivo ATmegaBOOT.hex:**

```
:101C000012C02CC02BC02AC029C028C027C026C0A3
:101C100025C024C023C022C021C020C01FC01EC0B8
:101C20001DC01CC01BC011241FBECFE5D4E0DEBF09
:101C3000CDBF10E0A0E6B0E0E6EAFFE102C005900B
```

```
:101C40000D92A236B107D9F711E0A2E6B0E001C0CB
:101C50001D92AA36B107E1F72BD0A3C1D1CF5D9B6E
:101C6000FECF8CB908955F9BFECF8CB108950F9382
:101C70001F93082F10E002C0F6DF1F5F1017E0F37C
:101C80001F910F9108951F93182FEDDF803231F4CB
:101C900084E1E5DF812FE3DF80E1E1DF1F9108953B
:101CA000E2DF803221F484E1DADF80E1D8DF0895D9
:101CB0000F931F93CF93DF93000010BC83E389B988
:101CC00088E18AB986E880BDBD9A1092680120E05B
:101CD00030E240E050E007C088B3832788BBCA01E8
:101CE0000197F1F72F5F2031B8F320936801BBDF34
:101CF000803381F1813399F4B6DF8032C1F784E11A
:101D0000AEDF81E4ACDF86E5AADF82E5A8DF80E212
:101D1000A6DF89E4A4DF83E5A2DF80E523C1803468
:101D200029F4A1DF8638B0F09EDF14C0813471F44D
:101D30009ADF803811F482E01DC1813811F481E00E
:101D400019C1823809F015C182E114C1823421F42D
:101D500084E18DDFA5DFCBCF853411F485E0F9CFA9
:101D60008035C1F38135B1F38235A1F3853539F47E
:101D70007ADF8093640077DF80936500EBCF863550
:101D800019F484E074DFF5C0843609F090C06BDF8D
:101D90008093670168DF809366018091690 18E7F7F
:101DA0008093690160DF853429F480916901816045
:101DB0008093690100E010E007C055DFF801EA599F
:101DC000FF4F80830F5F1F4F8091660190916701E5
:101DD0000817190790F347DF803209F088CF809108
:101DE000690180FF1FC000E010E014C0F801EA594B
:101DF000FF4F8091640090916500 6081C5D0809113
:101E000064009091650001969093650080936400 52
:101E10000F5F1F4F8091660190916701 08171907A6
:101E200028F343C0F894E199FECF1127E0916400B4
:101E3000F0916500EE0FFF1FC6E6D0E080916601CD
:101E40009091670180FF01C00196103051F422D0BB
:101E500003E000935700E8951DD001E1009357007F
:101E6000E8950990199016D001E000935700E89585
:101E70001395103258F011270DD005E0009357004C
:101E8000E89508D001E100935700E8953296029753
:101E900039F0DBCF009157000170013 0D9F308957C
:101EA000103011F00296E7CF112484E15BC0843733
:101EB00009F04BC0D8DE80936701D5DE80936601C0
:101EC000D2DE90916901853421F49160909369018B
:101ED0000DC09E7F9093690180916400 9091650090
:101EE000880F991F9093650080936400BCDE803258
:101EF00009F0FDCE84E1B3DE00E010E01EC0809169
:101F0000690180FF06C0809164009091650034D023
```

```
:101F100008C081FD07C0E0916400F0916500E49184
:101F20008E2F9DDE8091640090916500019690 93C4
:101F3000650080936400 0F5F1F4F80916601909150
:101F4000670108171907D8F20EC0853779F48BDEC0
:101F5000803209F0CCCE84E182DE8EE180DE83E93E
:101F60007EDE87E07CDE80E17ADEC1CE863709F056
:101F7000BECE80E088DEBBCEE199FECF9FBB8EBB9C
:101F8000E09A99278DB30895262FE199FECF9FBB44
:101F90008EBB2DBB0FB6F894E29AE19A0FBE019664
:061FA0000895F894FFCF44
:021FA6008000B9
:0400000300001C00DD
:00000001FF
```

```
:107DB000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFD3
:107DC000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC3
:107DD000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFB3
:107DE000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFA3
:107DF000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF93
:107E0000112484B714BE81FFF0D085E080938100F7
:107E100082E08093C00088E18093C10086E0809377
:107E2000C20080E18093C4008EE0C9D0259A86E02C
:107E300020E33CEF91E0309385002093840096BBD3
:107E4000B09BFECF1D9AA8958150A9F7CC24DD24C4
:107E500088248394B5E0AB2EA1E19A2EF3E0BF2EE7
:107E6000A2D0813461F49FD0082FAFD0023811F036
:107E7000013811F484E001C083E08DD089C08234E0
:107E800011F484E103C0853419F485E0A6D080C0E4
:107E9000853579F488D0E82EFF2485D0082F10E0AE
:107EA000102F00270E291F29000F111F8ED06801E7
:107EB0006FC0863521F484E090D080E0DECF843638
:107EC00009F040C070D06FD0082F6DD080E0C81688
:107ED00080E7D80618F4F601B7BEE895C0E0D1E017
:107EE00062D089930C17E1F7F0E0CF16F0E7DF06D8
:107EF00018F0F601B7BEE89568D007B600FCFDCFD4
:107F0000A601A0E0B1E02C9130E011968C91119780
:107F100090E0982F8827822B932B1296FA010C0160
:107F200087BEE89511244E5F5F4FF1E0A038BF0790
:107F300051F7F601A7BEE89507B600FCFDCF97BE46
:107F4000E89526C08437B1F42ED02DD0F82E2BD052
:107F50003CD0F601EF2C8F010F5F1F4F84911BD097
:107F6000EA94F801C1F70894C11CD11CFA94CF0C13
:107F7000D11C0EC0853739F428D08EE10CD085E9AC
:107F80000AD08FE07ACF813511F488E018D01DD067
:107F900080E101D065CF982F8091C00085FFFCCF94
:107FA0009093C60008958091C00087FFFCCF809118
:107FB000C00084FD01C0A8958091C6000895E0E648
:107FC000F0E098E1908380830895EDDF803219F02E
:107FD00088E0F5DFFFCF84E1DECF1F93182FE3DFCA
:107FE0001150E9F7F2DF1F91089580E0E8DFEE27F6
:107FF000FF270994FFFFFFFFFFFFFFFFFFFF0404C0
:00000001FF
```

En "negritas" está el Bootloader original del Atmega328p de Arduino. Es el que debe de copiar en un archivo de **Block de Notas** y salvarlo con extensión **.HEX**