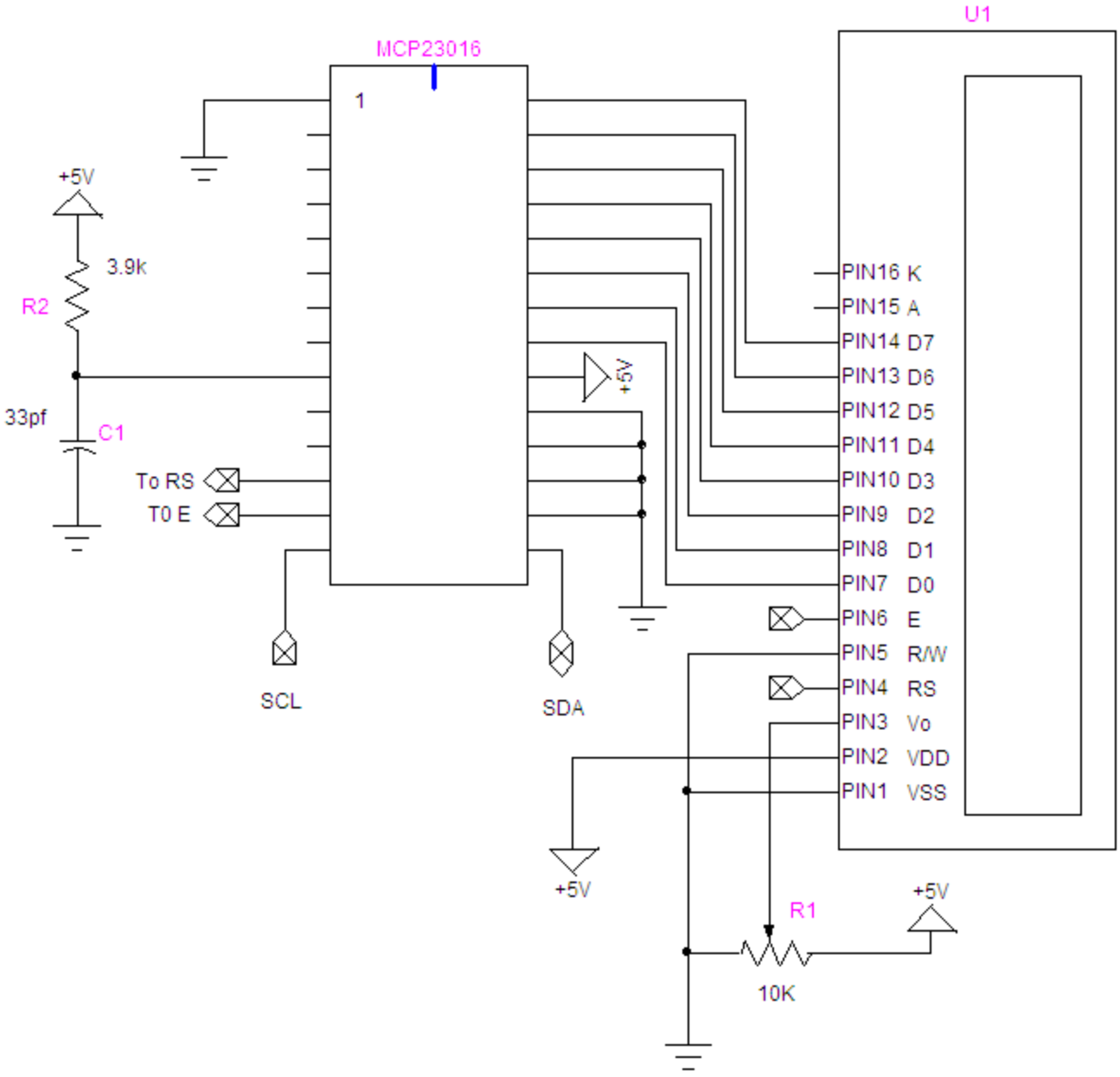



**Basic diagram.**

## Connecting the ATMEGA168/Arduino to MCP23016 and LCD Display

by Lewis Loflin

The address of the MCP23016 is 0x20.

**The data or command is written to port 0 (GP0) of the MCP23016**

**Note: RS register select pin 4 on display is connected to MCP23016 pin 13 (Gp1.7). 0 for command, 1 for data R/W of display goes to ground. C1 is 39pf and R2 is 3.9K. The MCP23016 pin 12 (Gp1.6) is E pin 6 on display.**

**E = connected display pin 6 (E), hi to lo transition will clock info into display. Register for command or data is selected by RS. Data is output in port0, two bits of port1 on the MCP23060 (bits 6, 7) are to control E and RS. R/W goes to ground.**

Hd44780 display commands:
0x0f = initiate display cursor on blinking
0x0c = initiate display cursor off
0x01 = clear display fills display with spaces (0x20).
0x02 = HOME returns to line one first character
0x38 = 2 lines X 16 char 8 bits mode. Defaults to 1 line mode.
0x10 = cursor left
0x14 = cursor right
0x18 = Shifts entire display left
0x1c = Shifts entire display right

Explanation of subroutines:

void BS() - backspace cursor on display. No return value.

void CLR() - clears display. No return value.

void type(char c) - outputs a character to display wherever cursor is at. Returns no value.

void writeCommand(char c) - will execute single command listed above. For example to clear the display to set the 2 line mode:

writeCommand(0x38); // two line mode
writeComaand(0x02); // clear display

**More sample code:**

```
 void typeln(char *array1, int i) - type one line of text to
display.
 his can be a quoted text string or array of characters while i
 selects line to be written to. The value of i is '1' for line
one
 and '2' for line 2. All text strings or arrays must have a "\n"
to
 terminate. Maximum 16 characters plus the '\n'.
 Note the "\n" not needed in the 0017 compiler.
```

The line:

```
for (int j = 0; (array1[j] != '\n') && (j < 16); j++)
type(array1[j]);
```

Has been modified to:

```
for (int j = 0; (array1[j] != 0) && (j < 16); j++)
type(array1[j]);
```

The following examples go in the void loop() {  )

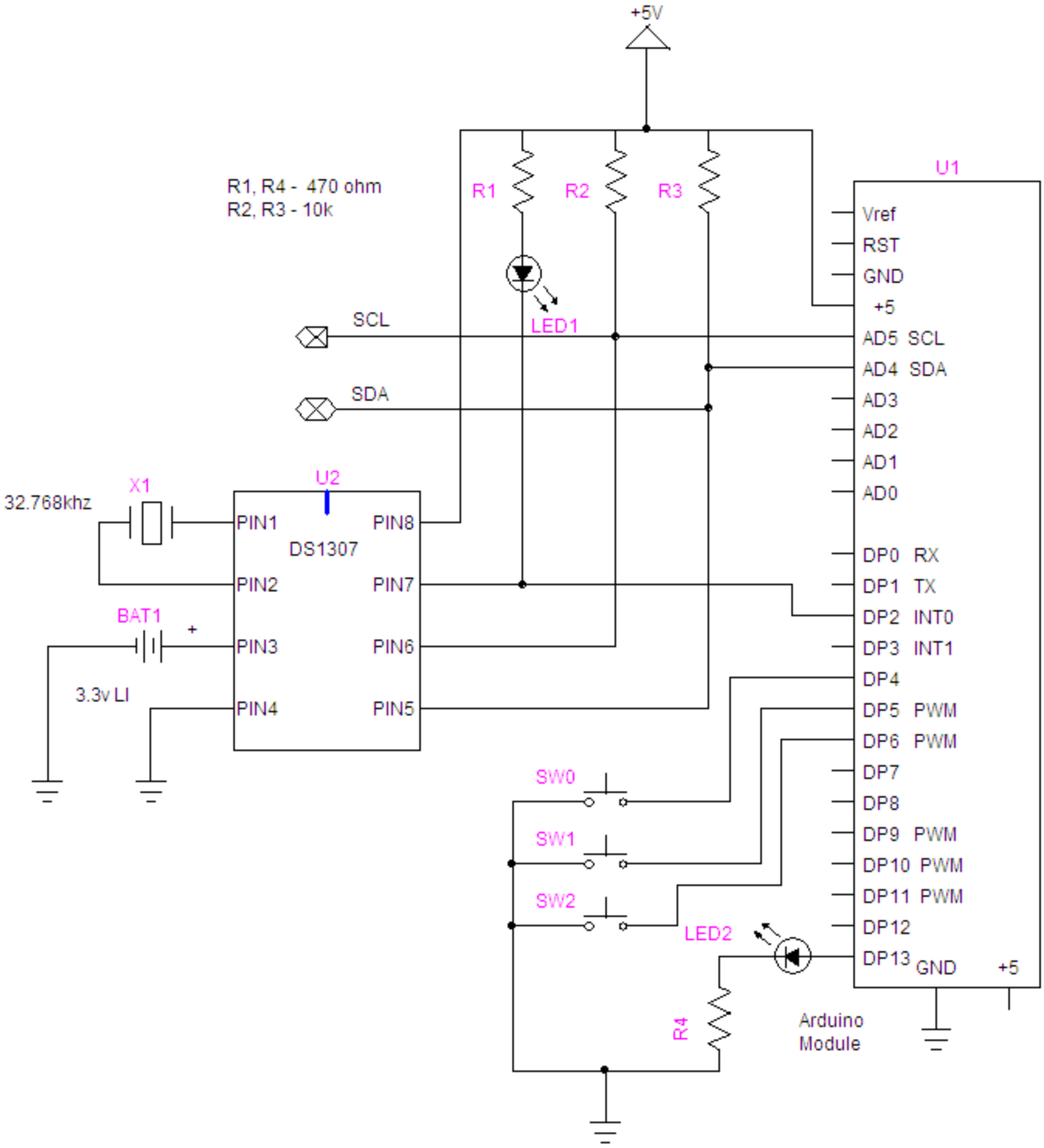
```
 Example 1:
 typeln("Hello word. \n", 1); // displays "Hello world." on line
one.

 Example 2:
 char textArray[] = "Hello world.\n";
 typeln(textArray, 2); // displays "Hello world." on line 2.

 Example 3 Convert int number to string and output to display.
 int intNumber = 1023; // the inter number range is from -32767
to 32767.
 char ST1[10]; //char array or string
 itoa(intNumber, ST1, 10); // covert an integer number to text
string.
 typeln(ST1, 1); // displays 1023 on line 1.


 Example 4 using PString.h from
http://arduiniana.org/libraries/pstring/
 The ZIP file must be downloaded and the unzipped folder
(PSstring) must
 be placed in My Documents/arduino-0017/hardware/libraries/
 Restart the compiler!
```

Be sure use PSring.h under Wire.h.

```
 PString has all the functions of print but instead of the
serial port
 will print to a text buffer/array. This can then be output with
typeln().

 Ex. 4A: Will print PI
  float PI = 3.141596;
  char buffer[16];
  PString(buffer, sizeof(buffer), pi);  // PI as text into
buffer
  typeln(buffer, 2); // prints 3.14 on line 2
```

```
  Ex. 4B:
  char buffer[16];
  PString(buffer, sizeof(buffer), "Hello World!");
  typeln(buffer, 2); // prints Hello World! on line 2

  Ex. 4C:
  char buffer[16];
  int SECS = 255;
  PString(buffer, sizeof(buffer), SECS);
  typeln(buffer, 2); // will print 255 on line 2


  Ex. 4D:
  char buffer[16];
  int SECS = 255;
  PString(buffer, sizeof(buffer), SECS, HEX);
  typeln(buffer, 2); // will print FF (HEX for 255) on line 2

  Ex. 4E:
  char buffer[16];
  int SECS = 255;
  PString(buffer, sizeof(buffer), SECS, OCT);
  typeln(buffer, 2); // will print 377 (OCT for 255) on line 2

  Ex. 4F:
  char buffer[16];
  int SECS = 255;
  PString(buffer, sizeof(buffer), SECS, BIN);
  typeln(buffer, 2); // will print 11111111 (binary for 255) on
line 2

  Ex. 5:
  float Pi = 3.1415;
  char buffer[16];
  PString str(buffer, sizeof(buffer));
  str.print("Pi = ");
  str.print(Pi);
  typeln(buffer, 2); // will print Pi = 3.14 on display line 2

  Ex. 6:
  float Pi = 3.1415;
  char buffer[16];
  PString str(buffer, sizeof(buffer));
  str.print("Pi = ");
  str.print(Pi);
  typeln(buffer, 1); // will print "Pi = 3.14" on display line 1
```

```
str.begin(); // re-use buffer
str.print("Length is ");
str.print(str.length());
typeln(buffer, 2); will display "Length = 10" in line 2

Ex. 7:
char buffer[16];
PString str(buffer, sizeof(buffer));
str.print("Capacity is ");
str.print(str.capacity());
typeln(buffer, 2); // prints "Capacity is 16" in line 2.
```




**Where to connect the MCP23016.**

Related:

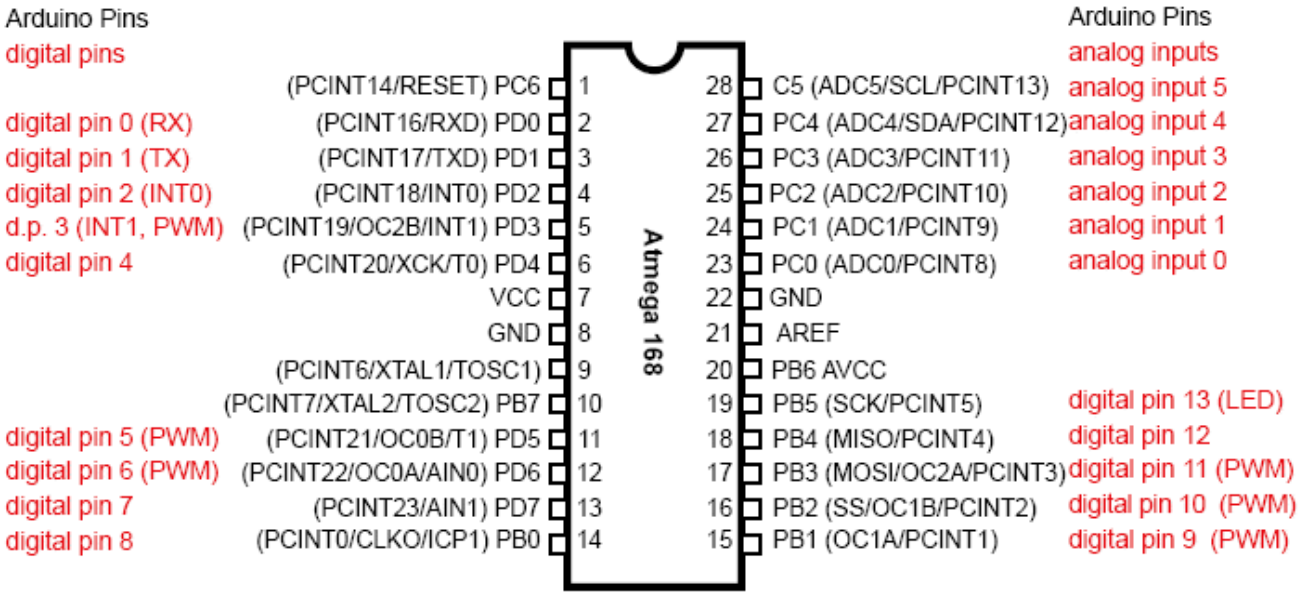
- Interfacing the ATMEGA168/Arduino to the MCP23016 I/O Expander
- Hitachi HD44780 Liquid Crystal Display PDF file
- MCP23016 I2C I/O Expander PDF file
- Bare Bone Kit Manual PDF file

```
#include <Wire.h> // specify use of Wire.h library.
#include <PString.h> // specify use of PSring.h library.


#define SW0  4
#define SW1  5
#define ledPin  13


// names can't have number as first char.
const byte HOME = 0x02;
const byte CL = 0x10; // cursor left
const byte CR = 0x14; // cursor right
const byte SL = 0x18; // Shifts entire display left
const byte SR = 0x1c; // Shifts entire display right


int j;

void setup()

{
  pinMode(SW0, INPUT);  // for this use a slide switch
  pinMode(SW1, INPUT);  // N.O. push button switch
  pinMode(ledPin, OUTPUT);
  digitalWrite(SW0, HIGH); // pull-ups on
  digitalWrite(SW1, HIGH);

  Wire.begin();
  Wire.beginTransmission(0x20);  // set mcp23016 output
  Wire.send(0x06);
  Wire.send(0x00);  // DDR Port0 all output
  Wire.send(0x0F);  // DDR 0-3 input 4-7 output
  Wire.endTransmission();
  // setup port 1 D7 = E; D6 = RS
  Wire.beginTransmission(0x20);
  Wire.send(0x01); //pointer
  Wire.send(B10000000); // setup for command mode
```

```
  Wire.endTransmission();


  writeCommand(0x38); // 2 lines
  writeCommand(0x0F); // blinking cursor
  CLR(); // clear dispaly
  writeCommand(HOME);



}



void loop()    {


  CLR(); // clear display

  typeln("Hello Lewis!\n", 1); // write to display line 1
  typeln("You got it! \n", 2); // write to display line 2
  toggle(ledPin);
  delay(500);
} // end loop


// Below we pass a pointer to array1[0]. If no '\n' then the lim
it is 16 char. (0 - 15)
void typeln(char *array1, int i)    {
  delayMicroseconds(1000);
  if (i == 1) writeCommand(0x80); // begin on 1st line
  if (i == 2) writeCommand(0x80 + 0x40); // begin on 2nd line
  for (int j = 0; (array1[j] != '\n') && (j < 16); j++)
type(array1[j]);
}




// send command to HD44780 display
//  E High to Low transition write command or data to HD44780 di
splay
//  RS 0 for command, default 1 for data
//  setup port1 D7 = E; D6 = RS
void writeCommand(byte x)    {

  Wire.beginTransmission(0x20);  //  mcp23016
```

```
  Wire.send(0x00); // begin here
  Wire.send(x);  // command code
  Wire.endTransmission();

  Wire.beginTransmission(0x20);  //  mcp23016
  Wire.send(0x01); // pointer
  Wire.send(B10000000);  // command mode E high
  Wire.endTransmission();

  Wire.beginTransmission(0x20);  //  mcp23016
  Wire.send(0x01); // pointer
  Wire.send(B00000000);  // char mode E low
  Wire.endTransmission();

  delayMicroseconds(100);

  Wire.beginTransmission(0x20);  //  mcp23016
  Wire.send(0x01); //  pointer
  Wire.send(B10000000);  // E high back to command mode
  Wire.endTransmission();


}

//  send char to HD44780 display
//  E High to Low transition write command or data to HD44780 di
splay
//  RS 0 for command, default 1 for data
//  setup port1 D7 = E; D6 = RS

void type(byte x)    {

  Wire.beginTransmission(0x20);  //  mcp23016
  Wire.send(0x00); // begin here
  Wire.send(x);  // data
  Wire.endTransmission();

  Wire.beginTransmission(0x20);  //  mcp23016
  Wire.send(0x01); // pointer
  Wire.send(B11000000);  // char mode E high
  Wire.endTransmission();

  Wire.beginTransmission(0x20);  //  mcp23016
  Wire.send(0x01); // pointer
  Wire.send(B01000000);  // char mode E low
  Wire.endTransmission();
```

```
  delayMicroseconds(100);

  Wire.beginTransmission(0x20);  //  mcp23016
  Wire.send(0x01); //  pointer
  Wire.send(B10000000);  // E high back to command mode
  Wire.endTransmission();


}


// clear display
void  CLR()   {
  writeCommand(0x01); // clear
}


// toggle the state on a pin
void toggle(int pinNum)
{
  int pinState = digitalRead(pinNum);
  pinState = !pinState;
  digitalWrite(pinNum, pinState);
}
```




**Pin mapping of the Atmega168 chip**

- More Arduino Projects
- Basic Transistor Driver Circuits for Micro-Controllers
- Opto-Isolated Transistor Drivers for Micro-Controllers

Arduino demos:

- Connecting the ATMEGA168/Arduino to MCP23016 and LCD Display
- Display Time/Date with Arduino, LCD Display, and DS1307 RTC
- Micro-controller AC Power Control Using Interrupts
- Controlling Low-Voltage Driveway Lights with the ATMEGA168/Arduino
- Hatching Chicken Eggs with ATMEGA168/Arduino
- The TSL230R Light to Frequency Converter and Arduino/ATMEGA168
- Interfacing the ATMEGA168/Arduino to the MCP23016 I/O Expander
- Using the ATMEGA168/Arduino with a DS1307 Real Time Clock
- Using a Unipolar Stepper Motor with a Arduino Micro-controller
- Using the ATMEGA168/Arduino with the TA8050 Motor Controller
- Using the ATMEGA168/Arduino with a 24LC08 Serial EEPROM
- Hardware Interrupts Demo and Tutorial for ATMEGA168/Arduino