Event counter with optical input

This article describes a tool for counting and controlling of motor rotations. Contract was to do a counter, where is possible to set a number of rotations and when motor finishes its job then a power for motor should be switched down. That motor drives a coil winder, so the speed is low.

Input is sensed with help of infrared parts. As output was selected LED display, because using was intended for dark work-room.

Counter displays numbers from -999 to 9999, up/down counting is controlled by optional input.

Counter contains five buttons, four for setting of desired number of rotations and one as start/stop button.

Function is following : man can set number of rotations, then press start button. Counter switches up a relay – motor runs and counter counts down. When counter reaches zero, then the relay is switched off – motor stops.

This version does not contain a 'count reset' button, you can simply add this functionality by connecting reset pin of MCU to GND.

SW

On web are presented many similar counters with one problem : output of phototransistor's collector is directly connected to input pin of MCU with confidence that signal is binary. Yes, at first, I was also sure this is enough. It is not, mainly in case of low speed. I tried this solution but sometimes the counter took false signal as correct, mainly when motor shaft vibrated. A good solution was to think the signal is analogue. I use AD converter, when input signal is more than 4V (Supply voltage is 5V) then signal is high. When signal is less than 1V then signal is low. When signal is between 1V and 4V then… then input to next filter is destroyed. AD conversion runs in background. Program checks input values from ADC every 256us. If eight (configurable value, see FILTER_MASK definition) consecutive values are the same, then the value is marked as OK. So, consecutive values can be 00000000 or 11111111 in case of nice input signal. If analogue value is between 1V and 4V then value is set to 0b00000010 (0x02). In next step this value is compared with 0x00 and 0xFF, then is clear now that 'bad' value 0x02 is ignored. Of course, signals are shifted every 256us. Who do not understand my explanation – see here ☺ :

```
clock = clock << 1;
if (adc_clock > 200) clock++; //valid 1, add 1
else if (adc_clock < 50) clock = clock; //valid 0, add 0
else clock = 2; //forbidden range, type poor value 0b10

dir = dir << 1;
if (adc_dir > 200) dir++;
else if (adc_dir < 50) dir = dir;
else adc_dir = 2;

//if last 8(?) values were the same, then we could take the averaged value as correct info
if ( (dir == 0) || ((dir & FILTER_MASK ) == FILTER_MASK) ) dir_last = dir; //valid bits
```

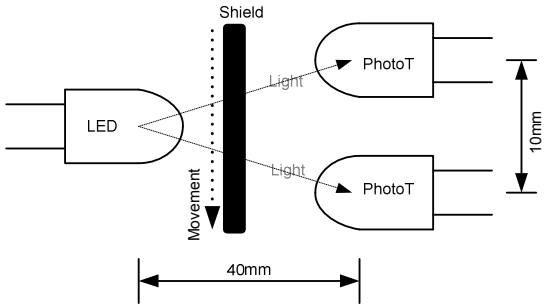
```
	if ( (clock == 0) || ((clock & FILTER_MASK) == FILTER_MASK) )
	{
		if ( clock_last > clock )
		{
			if (dir_last) main_counter--; //if DIR LED is not connected, then is internally connected to log.1
			else main_counter++;
		}
		clock_last = clock;
	}
```

ATmega8 runs from internal oscillator, frequency is 8MHz.

Code is written in C language, compiled under avr-gcc 3.4.6

HW

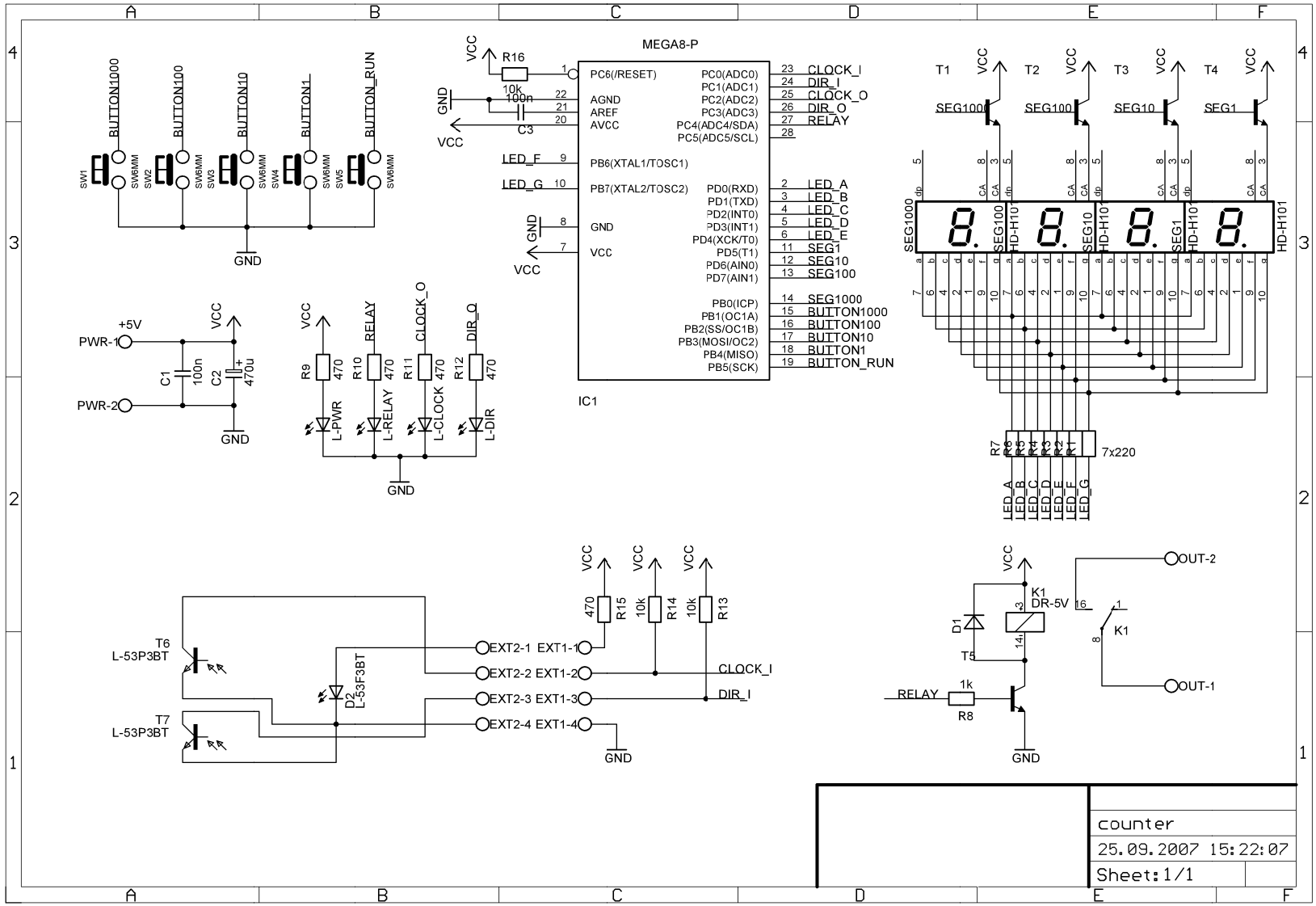
Default HW configuration consists of three infrared elements : one LED and two phototransistors. The values (R13, R14, R15) mentioned in schematic are fine for following layout :



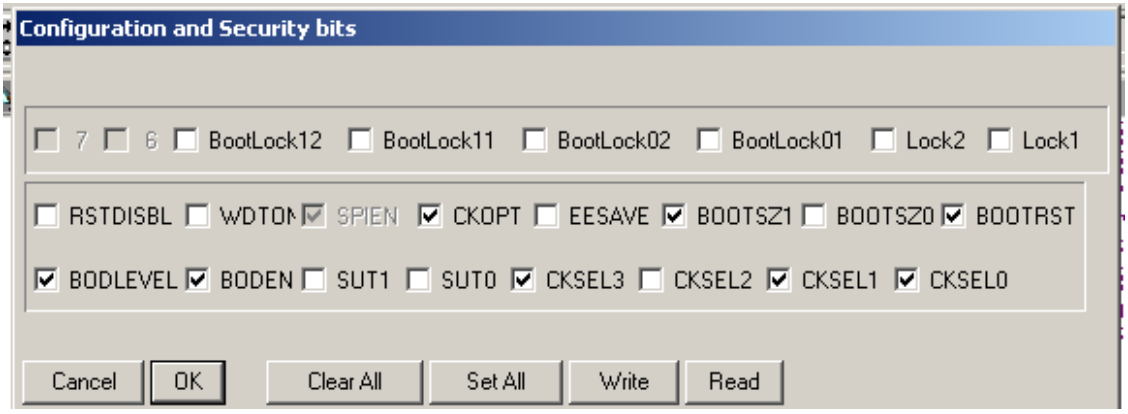
Types of LED and transistors are not critical, I think all similar parts will work, maybe with small change (of R13, R14 and R4), probably without.

A width of shield must hold darkness for both phototransistors at the same time, otherwise counter will work to one direction only.

If DIR phototransistor is not connected, then the counter counts down. Up-counting can be set by connecting EXT1-3 pin to ground.


MEGA8-P
IC1
BUTTON1000
BUTTON100
BUTTON10
BUTTON1
BUTTON_RUN
R16
10k
100n
C3
LED_F
LED_G
PC6(/RESET)
AGND
AREF
AVCC
PB6(XTAL1/TOSC1)
PB7(XTAL2/TOSC2)
GND
VCC
PC0(ADC0)
PC1(ADC1)
PC2(ADC2)
PC3(ADC3)
PC4(ADC4/SDA)
PC5(ADC5/SCL)
PD0(RXD)
PD1(TXD)
PD2(INT0)
PD3(INT1)
PD4(XCK/T0)
PD5(T1)
PD6(AIN0)
PD7(AIN1)
PB0(ICP)
PB1(OC1A)
PB2(SS/OC1B)
PB3(MOSI/OC2)
PB4(MISO)
PB5(SCK)
CLOCK_I
DIR_I
CLOCK_O
DIR_O
RELAY
LED_A
LED_B
LED_C
LED_D
LED_E
SEG1
SEG10
SEG100
SEG1000
BUTTON1000
BUTTON100
BUTTON10
BUTTON1
BUTTON_RUN
T1
T2
T3
T4
SEG1000
SEG100
SEG10
SEG1
HD-H101
R7
7x220
+5V
PWR-1
PWR-2
C1
100n
C2
470u
R9
R10
R11
R12
470
L-PWR
L-RELAY
L-CLOCK
L-DIR
T6
L-53P3BT
T7
L-53P3BT
D2
L-53F3BT
R15
R14
R13
10k
EXT2-1 EXT1-1
EXT2-2 EXT1-2
EXT2-3 EXT1-3
EXT2-4 EXT1-4
K1
DR-5V
D1
T5
RELAY
1k
R8
OUT-2
OUT-1
counter
25.09.2007 15:22:07
Sheet:1/1

Here is screenshot from PonyProg to explain fuse configuration



One tip: MCU has one free pin, if longer distance is necessary (few meters), then I think is possible to generate on this pin 36kHz and instead receiving phototransistor is possible to use any IR receiver (SFH5110, TSOP1736…)

Peter Rosko, p.eastman@zoznam.sk