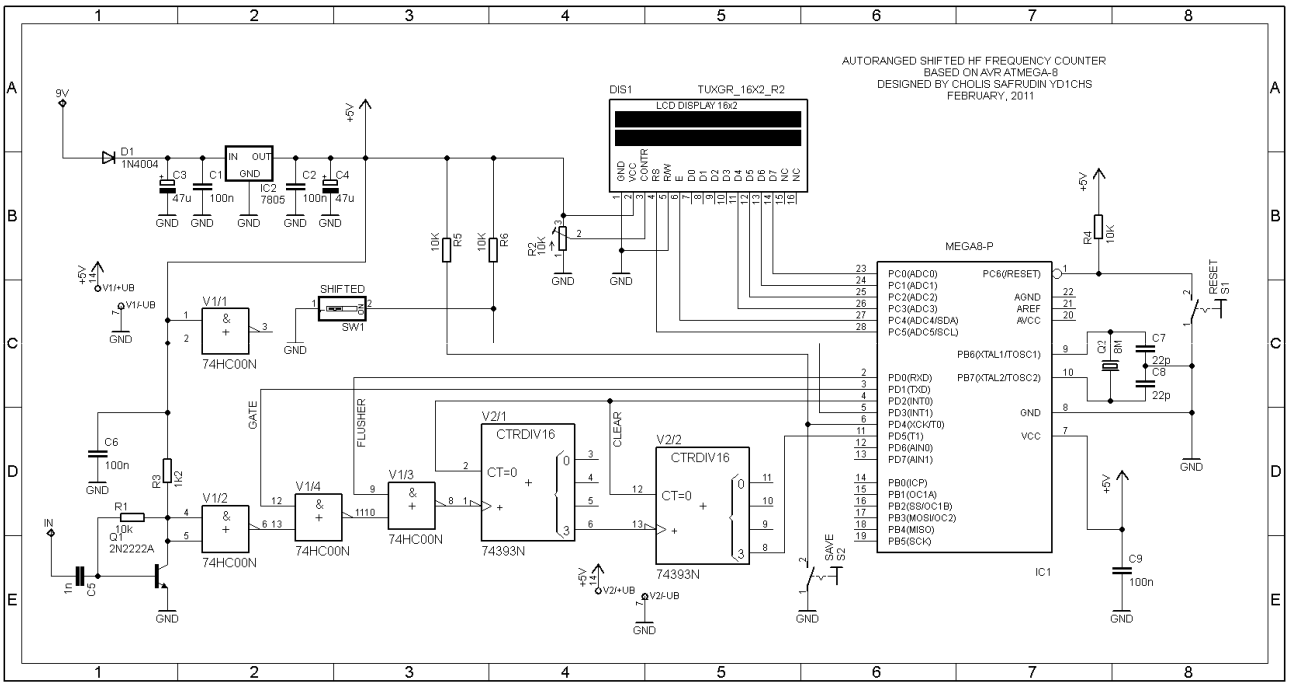
AUTORANGING SHIFTED FREQUENCY COUNTER BASED ON AVR ATMEGA-8

Designed by Cholis Safrudin YD1CHS
http://yd1chs.wordpress.com
February, 2011



Thanks to Google Translate …

ABSTRACT

Still rounding out of the 8 bits microcontroller world, but this time I will show the product of the AVR i.e. 28 pins ATmega-8. Project that I was designed is an old topic that is a frequency meter for HF bands that are equipped with autoranging and IF shifting facilities.

What I mean by autoranging feature is the FC will adjust the display frequency readings automatically, from xxx.xx Hertz, xxx.xxx.xx KHz and xx.xxx.xx MHz.

Furthermore, the Shifting IF feature is the display frequency that has been added or subtracted to an IF frequency which previously had been stored in the EEPROM. IF Frequency Selection made by the user by measuring the LO frequency in the block, then storing into the EEPROM, so any IF frequency can be used as a reference. In the IF shifted mode, the second row on the LCD will display additional information, namely:

- Shifted F=0
- Shifted F>IF
- Shifted F<IF, atau
- Shifted F=IF

If the user does not need this feature, then simply do release the "SHIFT" button. Thus, the user will be able to find out how much the IF frequency is stored in the EEPROM by comparing differences in the frequency display at both mode "Shifted" and "UN-Shifted".

To adapt the microcontroller in order to work up to the highest HF frequency (i.e. 30MHz), so I added an external prescaler before entering into the T1 port on the microcontroller. To get acceptable sensitivity to RF signals with small amplitude, the value of R3 (1k2) can be adjusted so that when there is no entry input, NAND gate circuit in an “almost active” position. Pre-Amp circuit can be replaced with other designs, but the “NAND Gate Port” and “Flusher GATE Port” should be maintained as in the original design.

Frankly, I have not done this design into real hardware experiment, but the design had been tried dan debugged using the help of Proteus demo software that can be downloaded for free on the Internet. Here are a few looks:

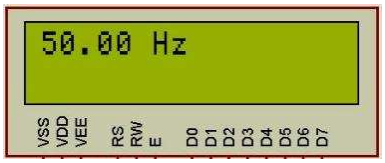
SIMULATION OF SYSTEM PERFORMANCE

Views on the side is a welcome message, really just to make sure the user that the basic components of a circuit which includes firmware, hardware and LCD have been working according to plan.

Further, I have been doing storage IF frequency of 50 Hertz and now its value has been stored safely in EEPROM. I will try to do the reading frequency for the 3 possibility condition, they are below the IF, the same as the IF as well as above the IF. And will we notice how their behaviour when making measurements on both the "UN-shifted" and "Shifted" modes.

The picture on the upper right is the result of reading an input signal measured at 5 Hertz with fashion "UN-shifted", shown readings in accordance with the frequency source.

Furthermore, the switch "SHIFT" is activated (pressed), so the current entry on the mode "Shifted" to the IF frequency as I mentioned at 50 Hertz. Looks interpretation is slightly different from the first reading, where the second line displayed a message for the user, namely a shifted mode and “F <IF”. The point of the “F <IF” is the measured frequency of 5 Hertz is smaller than the frequency of IF signal that is stored in the EEPROM that is equal to 50 Hertz.

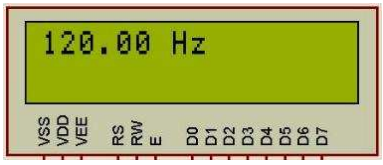


The next scenario is to set the input signal with frequency equal to the IF frequency, which is 50 Hertz. Images next to the left shows the input signal in question.

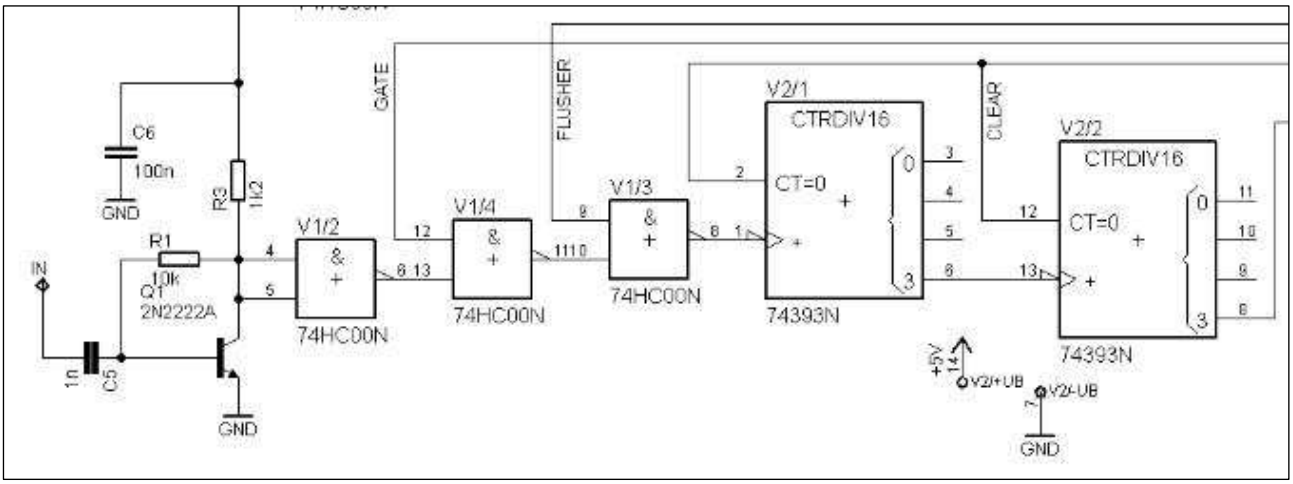
In mode "shifted" will be shown readings of 0 Hertz and additional information on the running mode, plus F = IF. Exactly with the previous explanation, that the frequency of the signal being measured together with fekuensi IF.

Lastly, what if the incoming signal has a frequency higher than the IF frequency and I think easily predictable result.

Input frequency of 120 Hertz, so that the mode "shifted" to 70 Hertz, plus readable description “F> IF”.



Well, I think there must be someone who ask why such experiment is done using Low Frequencies below 1kHz, the answer is a limitation of the simulator program and computer speed. To simulate the above 1kHz will decrease performance of simulation software.



HOW IS IT WORKED (HARDWARE)

The hardware circuit is divided into 4 parts, first is the RF Pre-Amp which is formed from a small signal transistor 2N2222A or other types. To get the effect of different sensitivity, this circuit can be modified, but ensure that the coupling to the next block does not use a capacitor coupling for the needs of a particular voltage on the input of NAND gate leading to the position just before declared as bit 1 (active) by the microcontroller.

The second part are three NAND GATEs (74LS00 or similar) connected to a serial configuration. So when the signal on NAND GATE GATE both activated (active low), then in accordance with the truth table of NAND GATE will be obtained on the condition 1, in accordance with the conditions of the input signal.

NAND GATE fourth function as Flusher, for flushing the rest of the clock that is resided in the buffer of the external prescaler. This particular NAND GATE also serves as the second GATE.

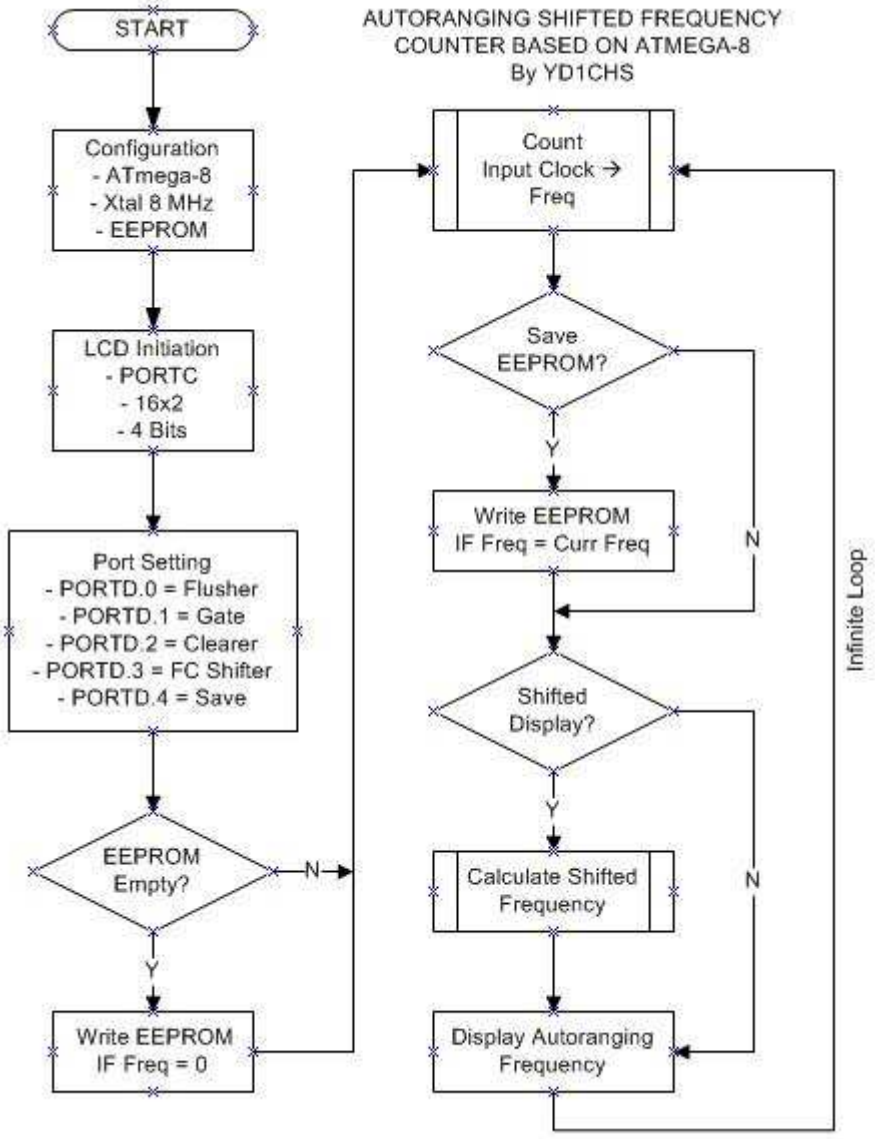
By controlling the activation of the GATE signal (active Low), then we can set when the input signal into the microcontroller and may be terminated at any time, including how long GATE should be opened, because the sampling period is associated with the CPU Speed and determine what level of accuracy of the counters that we were designed.

The third part is an external prescaler (74LS393) with a ratio of 1:256, which means that it takes 256 clock inputs to get 1 pc clock at the output side. Last position condition prescaler output bits are stored (latched) and can be waived when we reset pin, i.e. pin 2 and 12 are activated (active High).

The last section which is also the brains of this series is microcontroller ATmega AVR-8 that is connected to a 16x2 LCD.

HOW IS IT WORKED (SOFTWARE)

In a simple algorithm used in this application is shown by some of the following flowchart..



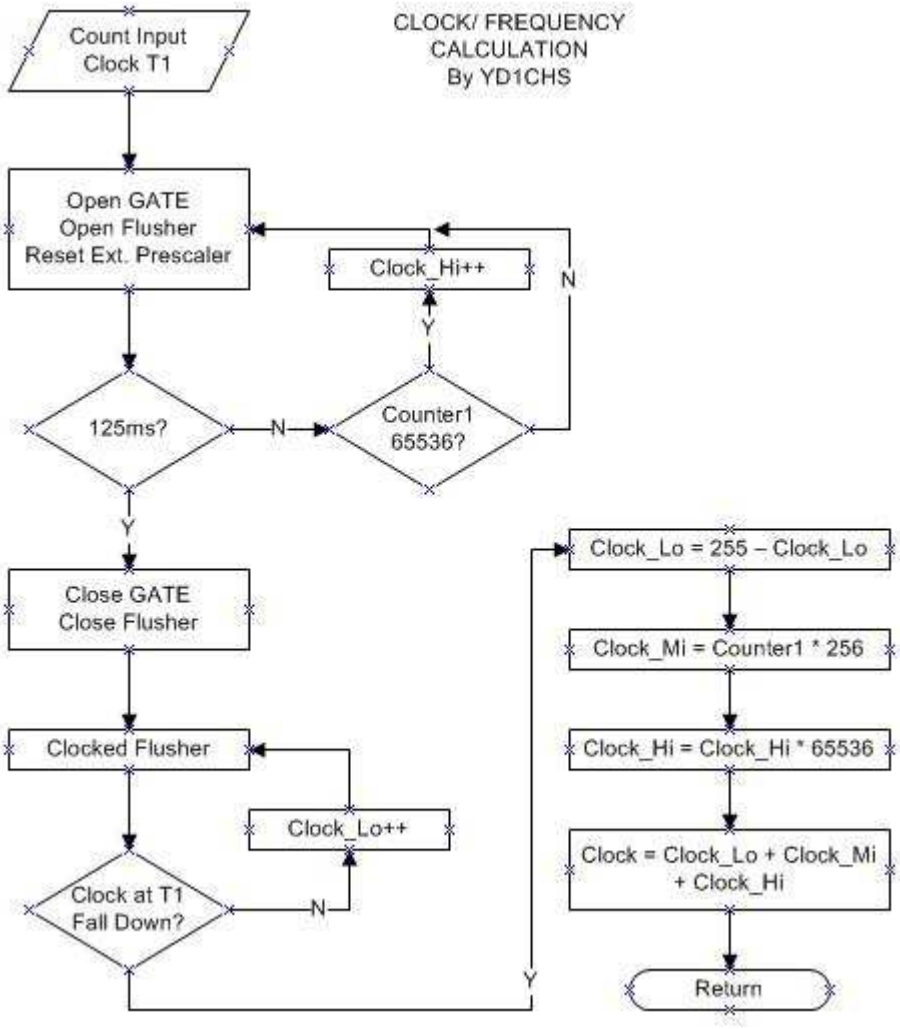
The flow chart in addition to the main process. Striped box on both sides means a process, and will be explained in more detail later.

As usual, the first step prior to major surgery done, then do some configuration and initiation of several things.

First is the machine configuration will be used, namely ATmega AVR-8 with an external source clock of 8MHz XTAL (16MHz maximum, I happened to not have a 16MHz XTAL, so in this design I use 8MHz). Because during the operation using the EEPROM, then we must tell the machine at the beginning of this memory usage.

Next step is initiation LCDs which include PORTC as a point of interfacing with LCD, using 16x2 LCD and uses a 4-wire communication mode (Data to Data to sd-4-7).

Some ports D declared as I / O. PORTD.0 used as a control signal to the NAND GATE Flusher third, PORTD.1 used as a control signal to the NAND GATE GATE second, while functioning as a controller signal PORTD.2 "CLEARER" or the signal to empty the content of external prescaler (74LS393). Two other PORTD are used as an input pin to be connected to the switch, namely PORTD.3 to enable the measurement mode (shifted or non-shifted) and PORTD.4 used to keep the IF frequency to the EEPROM. Subsequent operations were run in accordance with grooves on top.



In the major groove seen two processes are simplified with a picture box with a line on both sides. Figure disampaing left is one of them, namely the calculation algorithm of the clock of the input signal.

Clock entering the microcontroller is that has been through an external prescaler ratio of 1:256.

The second gate opened simultaneously for 125ms to pass the entrance to the microcontroller clock. 125ms election must be in accordance with CPU Speed of the microcontroller is used. ATmega 8MHz XTAL-8 with a CPU Speed of 8 MIPS (Mega Instructions Per Second). That is, every operation performed AVR microcontrollers only requires 1 clock cycle. Compare with PIC microcontroller product that has a CPU Speed clocknya quarter of the source, or in other words one operation requires four clock cycles!

With a period of 125ms then without using additional external components, microcontroller will be able to measure frequencies with an accuracy of 8 Hertz, but by using additional external prescaller plus a special algorithm to drain all the contents of buffer, so as to get the accuracy up to 1 Hertz!

Then how the upper limit of measurement? Since the AVR only need 1 clock cycles in one full operation, then it should be the maximum frequency that can be measured is the maximum amount of clock is used, in this case 8MHz. Why then why this series is claimed to work up to the range of HF, is not the highest in the definition of HF frequency is 30MHz? His response prescaller who do it for us.

In this design, the 74LS393 will do prescalling 1:256, so in theory this circuit capable of measuring up to 256 * 8MHz = 2GHz. But according to the datasheet 74LS393 only able to perform their duties until the frequency of 35MHz, so the number tersebutlah be the upper limit of measurement.

The second process is indicated with striped boxes on both sides are counting on the frequency of the IF Shift.

Chart shows besides simple and easiest method to get the effect of the shift, which is calculated by subtracting the clock time with the IF frequency has been stored in the EEPROM.

Furthermore, several grooves at the bottom is an operation to display a message on the second line of LCDs that are in fashion "shifted" and tells how to position F of the IF. From this simple information, if the circuit is applied together with a single transveicer Conversion, the user can use it to find work in USB or LSB.

Ifwe do not never make a stored value IF frequency ,then the system is designed to automatically write to the EEPROM IF frequency of 0 Hertz . Thus, when a user uses the mode "shifted" so he will be told that IF = 0.

HARDWARE REALIZATION

I am including the firmware from this project include files *. HEX (please do not ask me the firmware Such as *. ASM or other extentions) and ready to be downloaded to the microcontroller ATmega-8. Because everything is merely an experiment, so all the risk that exists is in your own hands.

In doing my downloads using software Khazama AVR Programmer with fuse bits settings blank (done by firmware settings).

USAGE PROCEDURE

Following the procedure the use of this project:

How to turn on:

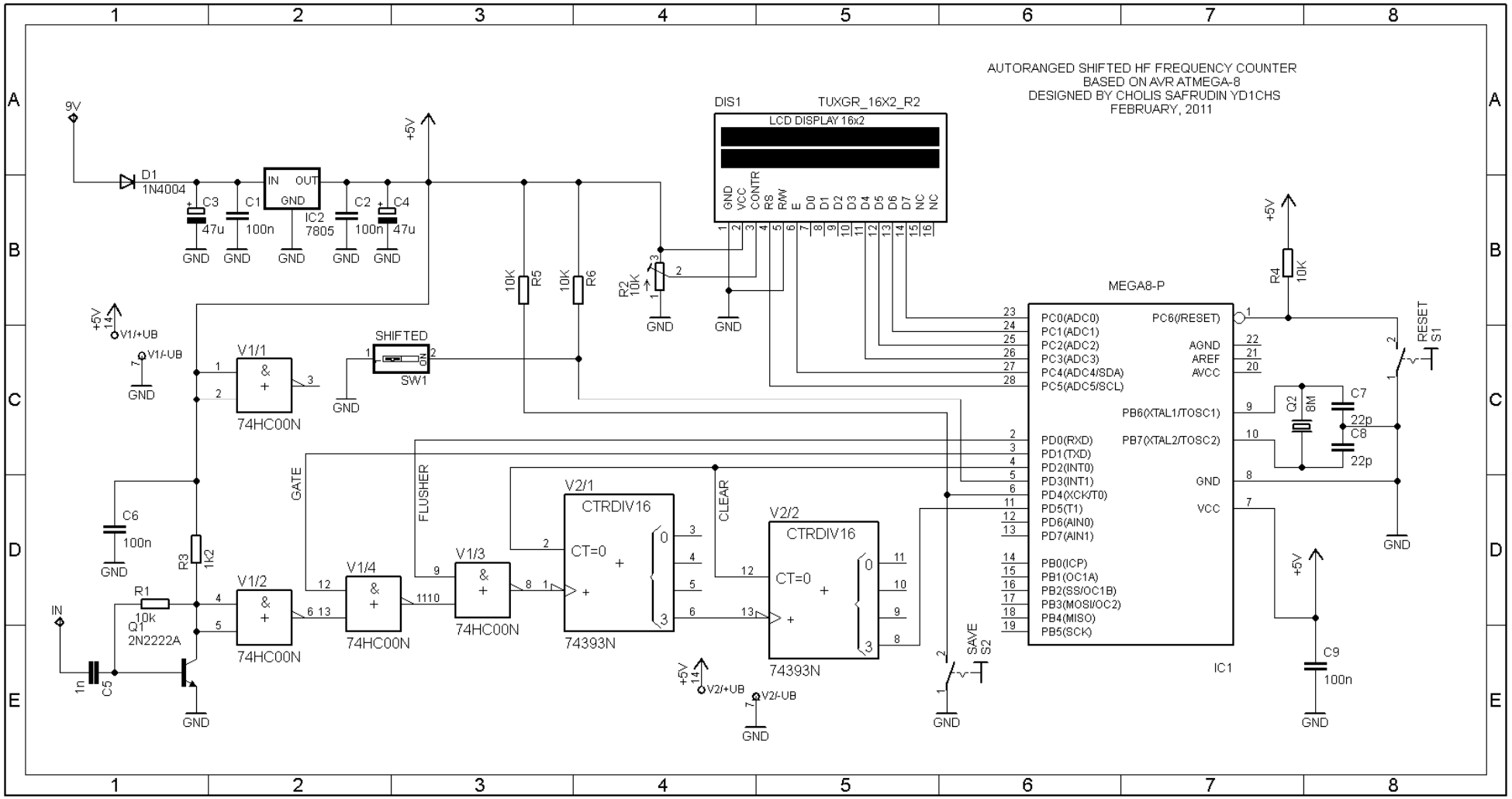
1. Make sure the whole set of connected properly and the whole set of ration voltage is 5V.
2. Be sure to operate the LCD backlight and contrast settings sufficient.
3. Make sure there is no connection active component of reverse polarization.
4. Connect to ration 9V battery and wait a while until the frequency display is shown.

How to save IF Frequency:

1. Make sure all the switches in the OFF (unpressed).
2. Follow the procedure above ignition.
3. In mode "UNSHIFTED" measuring the frequency of LO (IF frequency) is desired and wait until the reading stabilized.
4. Press the switch "SAVE" only once.
5. Press the switch "SHIFT" to measure the frequency LO modes shift toward the top, and the measurement should be close to 0 Hertz.
6. The series is ready for use.

ACKNOWLEDGMENTS

- Pre-Amp, NAND GATE and Pre-Scaller circuit is adopted from the work of E. Skelton EI9GQ.
- Design and programming side of the microcontroller firmware on ATmega-8 are the original works of Cholis Safrudin YD1CHS.


AUTORANGED SHIFTED HF FREQUENCY COUNTER
BASED ON AVR ATMEGA-8
DESIGNED BY CHOLIS SAFRUDIN YD1CHS
FEBRUARY, 2011
9V
D1
1N4004
C3
47u
C1
100n
IC2
7805
C2
100n
C4
47u
+5V
GND
DIS1
TUXGR_16X2_R2
LCD DISPLAY 16x2
GND VCC CONTR RS RW E D0 D1 D2 D3 D4 D5 D6 D7 NC NC
R5
10K
R6
10K
R2
10K
MEGA8-P
PC0(ADC0)
PC1(ADC1)
PC2(ADC2)
PC3(ADC3)
PC4(ADC4/SDA)
PC5(ADC5/SCL)
PC6(/RESET)
AGND
AREF
AVCC
PB6(XTAL1/TOSC1)
PB7(XTAL2/TOSC2)
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)
GND
VCC
IC1
R4
10K
RESET
S1
Q2
8M
C7
22p
C8
22p
C9
100n
V1/+UB
V1/-UB
V1/1
74HC00N
SHIFTED
SW1
GATE
FLUSHER
CLEAR
V2/1
CTRDIV16
74393N
V2/2
CTRDIV16
74393N
V1/2
74HC00N
V1/4
74HC00N
V1/3
74HC00N
C6
100n
R3
1k2
R1
10k
Q1
2N2222A
IN
C5
1n
V2/+UB
V2/-UB
SAVE
S2