AVR DDS signal generator V1.0

Document version V1.0

Introduction

Sometimes when tuning various electronic hardware we need simple signal generator with various waveform and frequency. One of the options is to by a professional with variable gain professional coating and many additional functions. But if you are an amateur you might want to build one. This small project is dedicated for building one of those signal generators.

Specification

AVR DDS signal generator consists of following parts:

- Atmel Atmega8 8 bit microcontroller;
- Supply source and voltage regulator;
- 2x16 standard LCD and shift register 74HC164;
- 7 buttons;
- R-2R resistor leader for DAC;
- Three outputs: universal(OUT) from DAC, PWM and pulses;
- Metal case;
- and microcontroller firmware.

Atmega8 microcontroller is a simplest of atmega series. There is 8kb of FLASH program memory, maximum frequency is 16MHz, which is used to reach DDS generators maximum resolution at maximum frequency. For know frequency is limited from **1 to 65535Hz** with minimal step of 1Hz.

DDS AVR generator is powered with 9V battery. Voltage is reduces to 5V and stabilized by 7805 voltage regulator.

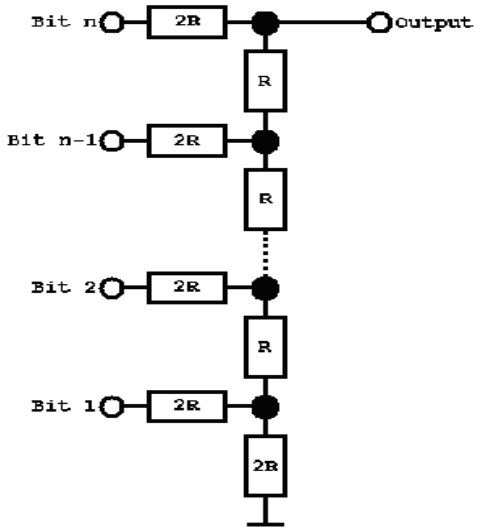
LCD is controlled using three wires through shift register 74HC164. So this register is used as serial to parallel converter in order to save microcontroller pins. LCD is controlled in 8 bit mode.

AVR DDS Generator uses 7 control buttons:

- Start;
- Stop, which is a reset also;
- IP- increasing value;

- DOWN- decreasing value;
- Mode1 – signal selection button;
- Mode2 – signal properties;
- Freq – signal frequency multiplier selection.

AVR DDS signal generator uses R-2R digital to analog (DAC) converter This is a simplest solution where resistors are connected in a ladder:



In this schematic R=10kohm. By using 8 bits and 5V step value is about 18.5mV. This is enough for getting average quality signals.

Generator has three outputs:

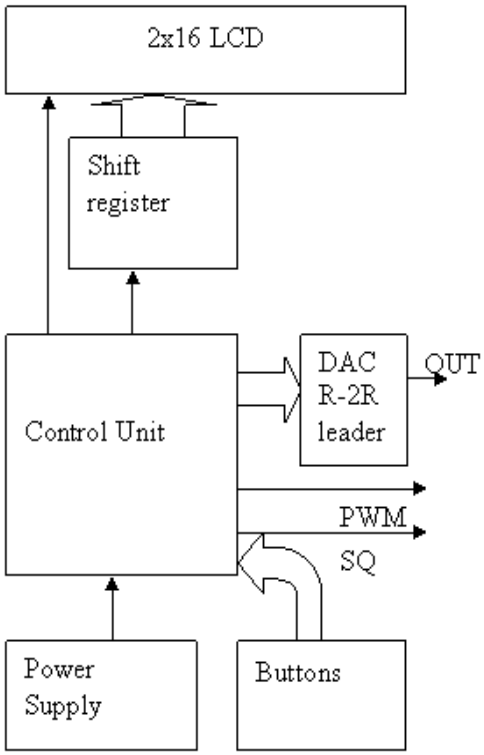
- Universal DAC output through R-2R ladder;
- PWM;
- Impulse (SQ);

Universal output (OUT) is a signal output from DAC. This output is to form various signals like sawtooth, sine, square, triangle.

PWM channel is used to form for PWM signal output – directly from timer.

SQ channel is additional channel to form square pulses or second PWM signal. Will be implemented in future.

Block chart of AVR DDS



In the block chart is described the structure of AVD DDS signal generator. You can se the signal paths. Each block part is described in specification.

Menu and program stages:

After generator is switched on the message is displayed in LCD:

			A	***V***	***R***		***S***	***I***	***G***	***N***	***A***	***L***			
	G	**E**	**N**	**E**	**R**	**A**	**T**	**O**	**R**		**V**	**1**	**.**	**0**	

Duration 1s.

R	***E***	***A***	***D***	***I***	***N***	***G***		***P***	***R***	***E***	***V***	***I***	***O***	***U***	***S***
	C	**O**	**N**	**F**	**I**	**G**	**U**	**R**	**A**	**T**	**I**	**O**	**N**		

After last signal generator configurations has been read it is displayed on LCD:
It last was square impulse used, then it shows:

O	***U***	***T***	_	|	‾	|	_						***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Signal can be changed with button “Mode1” by pressing it sequentially:
Above mentioned square signal.

O	***U***	***T***	|	/	|	/	|						***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Sawtooth signal

O	***U***	***T***	|	\	|	\	|						***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Reverse saw tooth

O	***U***	***T***	***/***	****	***/***	****	***/***	****					***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Triangle

O	***U***	***T***	**~**	**~**	**~**	**~**							***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Sine wave

O	***U***	***T***	**-**	***N***	***O***	***I***	***S***	***E***					***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Random signal

„Mode2“ button selects the step of frequency from 1 to 10000Hz by pressing it in series. “Up” and “Down” buttons change the frequency value by step value.
After “Start” button is pressed - last settings are saved in EEPROM and signal generation is started.
Above functionality is implemented and tested. In a future plans there are few more signal generation modes. By pressing “Mode1” you can see following screens whish may change a little during implementation:

P	***W***	***M***	**-**	***O***	***C***	***C***							***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

(Simple Output compare) using timer16

P	***W***	***M***	**-**	***S***	***I***	***N***	***M***	***D***	***S***				***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Sine wave modulated PWM (Dual Slope)

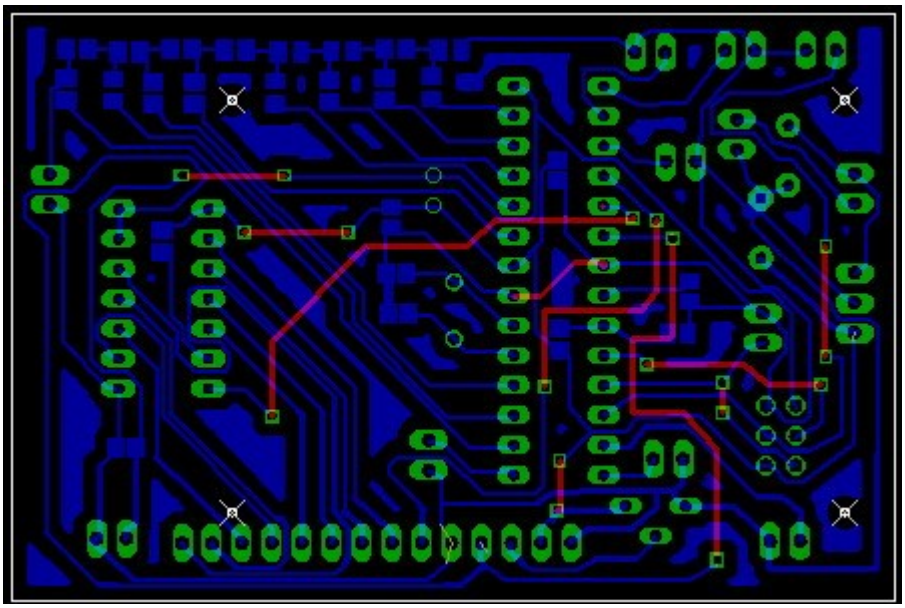
P	***W***	***M***	***S***	***Q***	***S***	***M***	***D***	***S***					***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Sine wave modulated PWM (Dual Slope)(PWM - positive polarity, SQ – negative polarity outputs)
P – positive, N – negative.

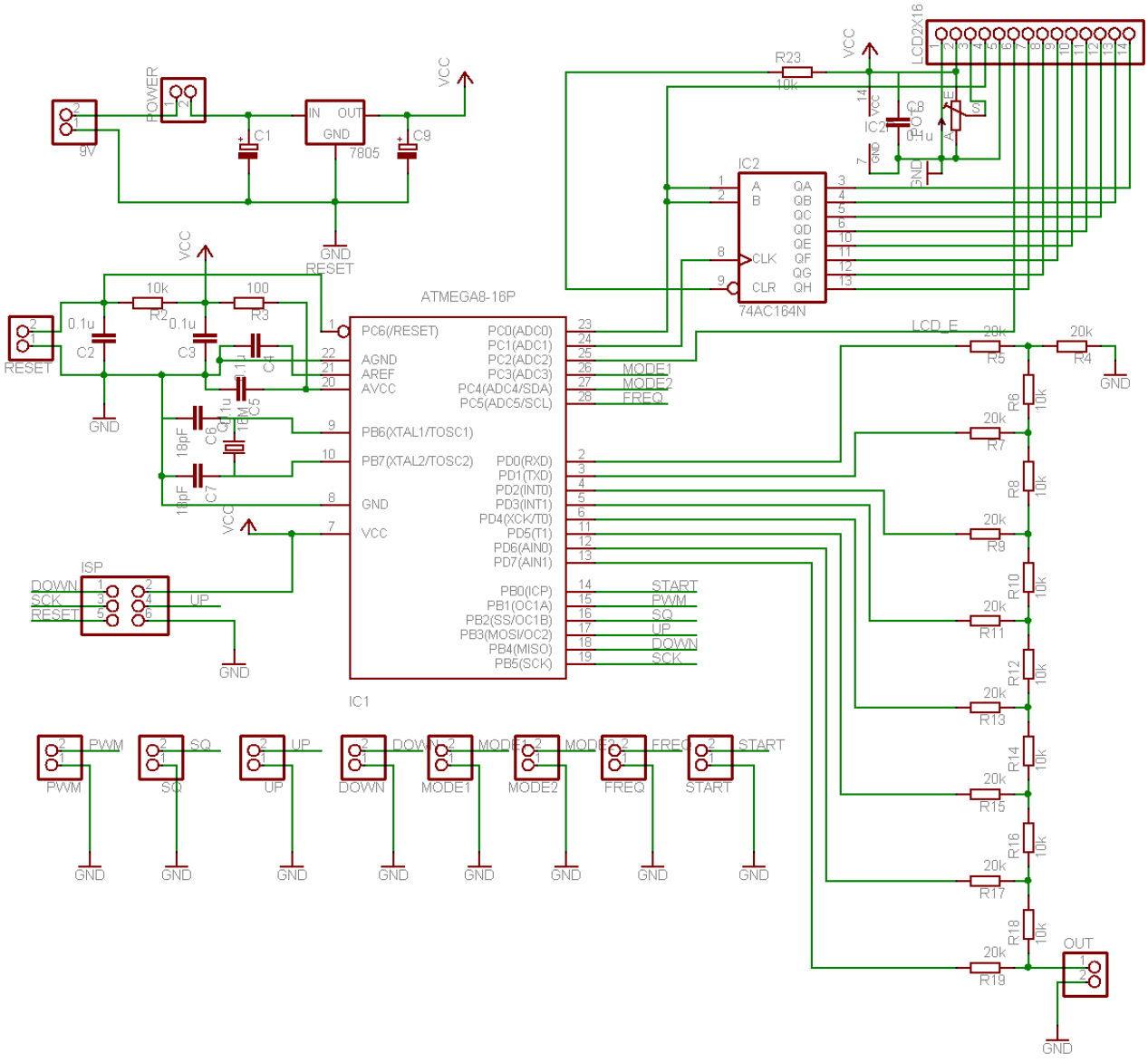
P	***W***	***M***	**-**	***C***	***M***	***D***	***S***		***5***	***0***	***%***		***O***	***F***	**F**
F	**R**	**E**	**Q**	**:**					**1**	**0**	**0**	**0**		**H**	**z**

Pulse width set by user PWM (Dual Slope). Setting is done by „Up“ and „Down“ buttons.

Circuit diagram and PCB



PCB is single sided with some wiring done in top side.



Circuit diagram explanation:

Microcontroller port pins are connected to:

- R-2R DAC is connected to D port pins 0...7;
- Shift register data line is connected to C port's 0 pin;
- Shift registers synchronize line is connected to port C pin ;
- LCD screen's E signal is controlled by C ports pin 2;
- Start button is connected to B port pin 0;
- Stop button is reset button;
- Up button is connected to port B pin 3;
- Down button is connected to port B pin 4;
- Mode1 button is connected to C port pin 3;
- Mode2 button is connected to C port pin 4;
- Freq button is connected to C port pin 5;
- SQ signal is coming from AVR b port pin 2;
- PWM signal comes out from Port B pin 1.

Part list (exported from eagle):

Qty	Value	Device	Parts
1		78MXXL	7805
2		CPOL-EUE5-4	C1, C9
12		PINHD-1X2	9V, DOWN, FREQ, MODE1, MODE2, OUT, POWER, PWM, RESET, SQ, START, UP
1		PINHD-1X14	LCD2X16
1		PINHD-2X3	ISP
1		TRIM_EU-B64W	POT
5	0.1u	C-EUC0805	C2, C3, C4, C5, C8
8	10k	R-EU_M0805	R6, R8, R10, R12, R14, R16, R18, R23
1	10k	R-EU_V526-0	R2
1	16M	CRYTALHC49S	Q1
2	18pF	C-EUC0805	C6, C7
9	20k	R-EU_M0805	R4, R5, R7, R9, R11, R13, R15, R17, R19
1	74AC164N	74AC164N	IC2
1	100	R-EU_M0805	R3
1	ATMEGA8-16P	ATMEGA8-16P	IC1

Tools used

In developing this AVR DDS signal generator There were following tools used:

Software:

- Eagle Cadsoft;
- WinAVR 20060421;
- PonyProg.

Hardware tools:

- Parts (Atmega8, PCB coated with photo resistive layer, box,...);
- UV exposure unit;
- PCB Etcher;
- Developer, Etching solution;
- Solder
- Programmer AVR ISP;
- Other tools...

Few words about firmware

Program is written in C language and can be compiled with WinAVR20060421 toolset. Programming is done by using AVR ISP and PonyProg programming software.

Program flow in few words:

When AVR DDS generator is switched on:

- Initialization of LCD;
- Reading previous settings from EEPROM memory and displaying to LCD;
- Using buttons settings and signals can be changed. When “Start” button is pressed, new settings are saved to EEPROM and signal generation starts.
- Note: After power up you can press “Start” to start last saved signal generation immediately;
- Generator is stopped by pressing Stop button which resets the generator. After Reset generator again loads its lasts configuration from EEPROM memory and is ready to start new generation.

EEPROM memory stores following data:

Address	*Value*
0	Mode
1	Freq[7...0]
2	Freq[15...8]
3	Freq[23...16]
4	Duty [%]
5	Other reserved for future
6...	Other reserved for future.

Mode values:

- **0 – OUT_|¯|_;**
- **1 – OUT|/|/|;**
- **2 – OUT|\|\|;**
- **3 – OUT/\/\/\;**
- **4 – OUT~~~~;**

- **5 – OUT-NOISE;**
- **20 – PWM-OCC;**
- **21 – PWM-SINMDS;**
- **22 – PWMSQSMDS;**
- **23 – PWM-CMDS;**

Frequency value is divided to 3 EEPROM bytes. Because max theoretical frequency is 16MHz whis hexadecimal value is 0XF42400:

Freq[23...16]	Freq[16...8]	Freq[7...0]

Duty value is from 1 to 99 in percents.

Program Structure:

LCD_3w.h – LCD settings

LCD_3w.c it contains LCD control functions

- void sendByteToRegister(uint8_t);
- void LCDenableCommand(void);
- void LCDdisableCommand(void);
- void LCDenableData(void);
- void LCDdisableData(void);
- void LCDsendChar(uint8_t); //forms data ready to send to 74HC164
- void LCDsendCommand(uint8_t); //forms data ready to send to 74HC164
- void LCDinit(void);
- void LCDwritebyte(uint8_t, uint8_t);
- void LCDdefinechar(const uint8_t* ,uint8_t);
- void LCDclr(void);
- void LCDhome(void);
- void LCDstring(uint8_t*, uint8_t);
- void LCDGotoXY(uint8_t, uint8_t);

main.c – main program text where:

- signal tables are stored;
- messages to LCD are described
- EEPROM initialization is done;
- changing parameters are implemented;
- signal generation (using in-line ASM routines) is performed.

Testing

For now testing is done by using frequency counter and Oscilloscope. In a frequency range of 1 to 65535Hz works OK. There can be greater frequency range programmed but with less resolution. I think for audio equipment testing there is more than enough frequencies.

Discussion

As I mentioned earlier, this is a first trimmed version of AVR controlled generator. It has half of working functionality in many cases this can be enough. Main signals are generated and can be controlled. In a future the functionality can be expanded to add PWM generation using timers. Generator generates signals as they are – directly from DAC. There is no variable gain regulation. To be more interesting project there can be VGA amplifier used with feedback to control output signal voltage. The project source code is open to modify and expand functionality. If there will be some good modifications – feedback is always welcome.