

WAVEFORM GENERATION USING DAC 0080

BY
SUBATHRA S

This work is licensed under the Creative Commons Attribution-NonCommercial-Share Alike 2.5 India License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/2.5/in/deed.en> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

WAVEFORM GENERATION USING DAC 0800

OBJECTIVE

To generate square waveform, triangular waveform, saw-tooth waveform and sine waveform using DAC 0800.

APPARATUS REQUIRED

- 8085 Microprocessor (ALS-SDA-85m) kit.
- Dual DAC Interface.
- Cathode Ray Oscilloscope.
- Flat Ribbon Cable.
- Power Supply (+5, +12v, -12v).

DESCRIPTION

The dual technology interface consists of two DACs the outputs of which are converted to voltages using Op-amps. The voltage outputs 0 to 10v or +/-5v of the Op-amps are terminated in a 4 way reliamate connector +0v external connection. A 10v of stable voltage for DAC is obtained using LM723 regulator. A preset is provided to adjust the unipolar or bipolar output are selected by either J1 and J3 to J2 and J4 or by isolating the jumpers.

DAC 0800 is a monolithic. Its unique features are

1. Typical setting time of 100ns
2. If the full scale analog voltage is x volts, the smallest unit or the LSB $(001)_2$ is equivalent to $x/2^n$ volts. This is defined as resolution.
3. The MSB represents half of full scale value. i.e. $MSB(100)_2 = x/2$ volt, assuming 3bit DAC.
4. For full scale, output is equal to the value of the full-scale input minus the value of 1 LSB input signal.
Full scale output = Full scale value – 1 LSB
5. Has complementary current outputs.
6. Has two quadrant wide range multiplying capacity.
7. Different output voltage of $20 V_{pp}$ with simple resistor load.

DESIGN

DAC interface section comprises of (NOT in NIFC-06)

1. Input/ Output Decoding
The IC decoder 74LS138 and a NAND gate 74LS00, form the address decoding logic with address C0H, DAC1 is selected and with C8H DAC2 is selected.
2. DAC Conversion Circuit
DAC Conversion Circuit comprises 74LS273 latch, DAC 0800 and I to V convertor. The 8-bit data on the data bus is input to DAC0800, which gives the equal and complementing current output. The output voltage varies in steps of $10/256 = 40mV$.

INSTALLATION PROCEDURE

1. Connect P3 in 85m to the connected C1 on the interface using a 26 core flat cable. Care should be taken that pin 1 of P3 on the kit coincides with pin1 of the cable. Observe the notch on the cable connector.

2. Power connections:

Connect +5/+12v/-12v to the interface color codes of power connectors on the interface

- +5 - Orange/Blue /White
- GND - Black
- +12v - Red
- 12v - Green.

PROBLEM ANALYSIS

SQUARE WAVEFORM GENERATION

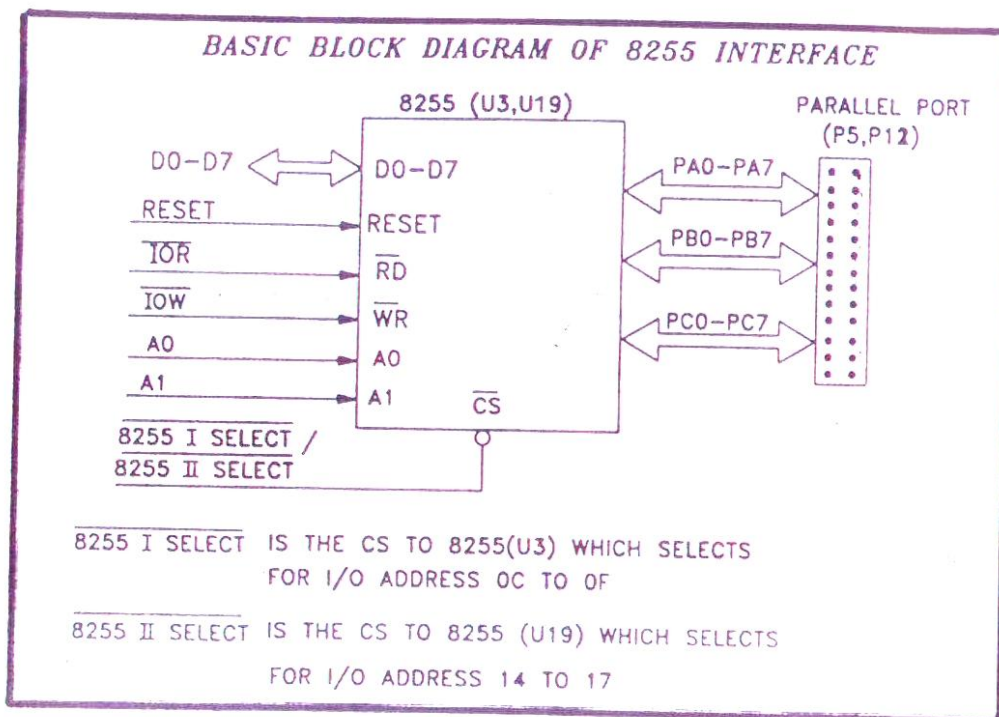
With 00_H as input to DAC, analog output is -5v and with FF_H as input, the output is +5v. Input 00_H and FF_H at regular intervals, to generate a square wave. The frequency can be varied by varying the time delay.

TRIANGULAR WAVEFORM GENERATION

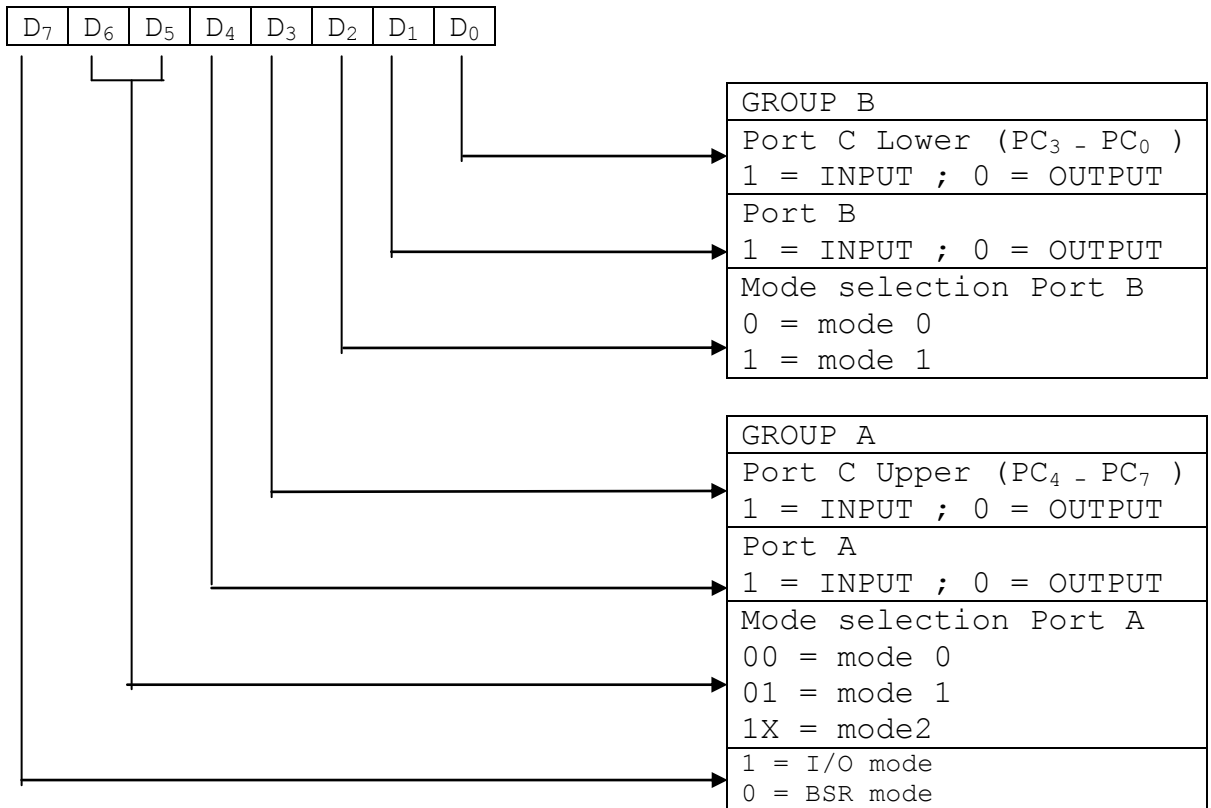
By outputting digital data from 00_H to FF_H in steps of 01_H and then from FF_H to 00_H. Decrementing in steps of 01_H, the triangular waveform is generated.

SINE WAVEFORM GENERATION

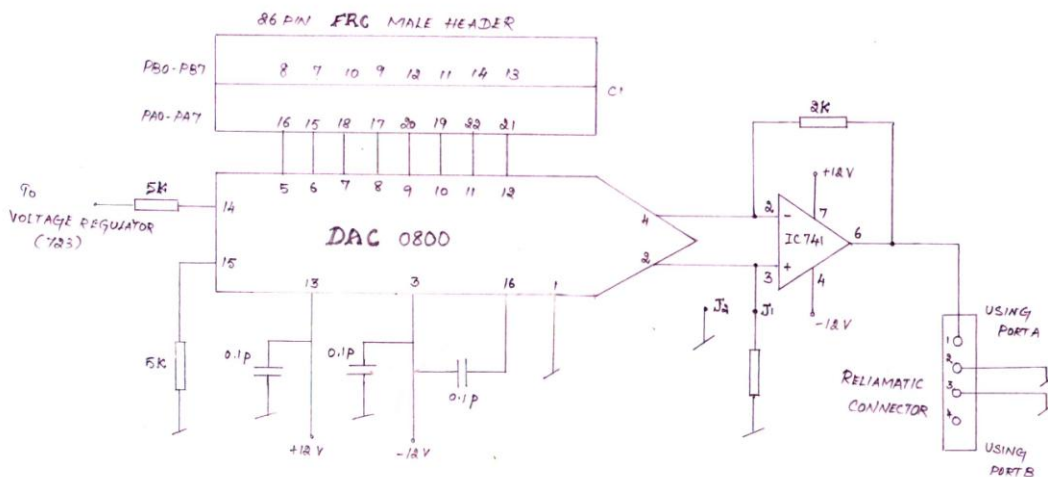
A look up table is formed and data are output continuously to DAC, from lookup table.
 $\alpha = 128 + 128 \sin \theta$ where $\theta = - 90$ degree to $+90$ degree.



CONTROL WORD 8255 PPI



CIRCUIT DIAGRAM OF DUAL DAC IN INFC-06 (or INFC-07)



SQUARE WAVEFORM GENERATION

ALGORITHM

1. Initialization a control word for 8255, for it to operate in I/O mode and for ports A, B and C to operate in output mode.
2. Clear the accumulator content and output it.
3. Call delay subroutine.
4. More immediate accumulator with FF_H and output it.
5. Continue the steps 2 to 4.

ASSEMBLY LANGUAGE PROGRAM

ADDRESS	LABEL	MNEMONICS	OPCODE/OPERAND
C800		MVI A, 80 _H	3E 80
C802		OUT CWR	D3 DB
C804	REPEAT	MVI A, 00 _H	3E 00
C806		OUT PORTA	D3 D8
C808		CALL DELAY	CD 15 C8
C80B		MVI A, FF _H	3E FF
C80D		OUT PORTA	D3 D8
C80F		CALL DELAY	CD 15 C8
C812		JMP REPEAT	C3 04 C8
C815	DELAY	MVI C, 85 _H	0E 85
C817	AGAIN	DCR C	0D
C818		JNZ AGAIN	C2 17 C8
C81B		RET	C9

PROGRAM TRACE

LABEL	MNEMONICS	DESCRIPTION																																											
	MVI A, 80 _H	Initializing the ports of the PPI 8255 as O/P ports by writing the control word as 80 _H . <table border="1" style="width: 100%; border-collapse: collapse; margin-bottom: 10px;"> <tr> <td style="text-align: center;">DATA BITS</td> <td style="text-align: center;">D₇</td> <td style="text-align: center;">D₆</td> <td style="text-align: center;">D₅</td> <td style="text-align: center;">D₄</td> <td style="text-align: center;">D₃</td> <td style="text-align: center;">D₂</td> <td style="text-align: center;">D₁</td> <td style="text-align: center;">D₀</td> </tr> <tr> <td></td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> <tr> <td style="text-align: center;">COMMENT</td> <td style="text-align: center;">I/O mode</td> <td style="text-align: center;">Mode0</td> <td style="text-align: center;">PortA O/P</td> <td style="text-align: center;">PortC Upper O/P</td> <td style="text-align: center;">Mode0</td> <td style="text-align: center;">PortB O/P</td> <td style="text-align: center;">PortC Lower O/P</td> <td></td> </tr> </table> 80 _H is moved to accumulator. REGISTERS <table style="margin-left: 20px;"> <tr> <td style="padding-right: 5px;">A</td> <td style="border: 1px solid black; padding: 2px 5px;">80</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="padding-left: 5px;">F</td> </tr> <tr> <td>B</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td>C</td> </tr> <tr> <td>D</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td>E</td> </tr> <tr> <td>H</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td>L</td> </tr> </table>	DATA BITS	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		1	0	0	0	0	0	0	0	COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P		A	80	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
DATA BITS	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																					
	1	0	0	0	0	0	0	0																																					
COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P																																						
A	80	XX	F																																										
B	XX	XX	C																																										
D	XX	XX	E																																										
H	XX	XX	L																																										
	OUT CWR	Control word specify the I/O function for each ports of 8255.																																											
REPEAT	MVI A, 00 _H	Initialize portA as 00 _H .																																											

		<p style="text-align: center;">REGISTERS</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="padding-right: 5px;">A</td> <td style="padding: 2px 5px;">00</td> <td style="padding: 2px 5px;">XX</td> <td style="padding-left: 5px;">F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>XX</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	A	00	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
A	00	XX	F															
B	XX	XX	C															
D	XX	XX	E															
H	XX	XX	L															
	<p>OUT PORTA</p>	<p>00_H is outputted thro portA. This is the beginning of the square waveform.</p> <div style="text-align: center;"> <p>AMPLITUDE (v)</p> <p>TIME PERIOD (ms)</p> </div>																
	<p>CALL DELAY</p>	<p>A small time interval is being introduced.i.e.the pointer moves a small distance.The delay ensures the frequency.</p> <p><i>NOTE: If you don't give any delay then you cannot see any square wave because of the speed at which the 8255 output line change state from high to low and low to high. Increase in delay to reduce the frequency and reduce the delay to increase the frequency.</i></p> <div style="text-align: center;"> <p>AMPLITUDE (v)</p> <p>TIME PERIOD (ms)</p> </div>																
	<p>MVI A, FF_H</p>	<p>FF_H is moved to accumulator. The pointer transfers its control from 00H to FF_H, which results in square waveform generation.</p> <p style="text-align: center;">REGISTERS</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="padding-right: 5px;">A</td> <td style="padding: 2px 5px;">FF</td> <td style="padding: 2px 5px;">XX</td> <td style="padding-left: 5px;">F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>XX</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	A	FF	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
A	FF	XX	F															
B	XX	XX	C															
D	XX	XX	E															
H	XX	XX	L															

	OUT PORTA	<p>FF_H is outputted thro portA. This is the ending of the square waveform.</p>																
	CALL DELAY	<p>A delay is introduced for visibility of transmission.</p>																
	JMP REPEAT	<p>The square waveform was generated continuously by repeating the cycle.</p>																
DELAY	MVI C,85 _H	<p>DELAY is a subprogram. Move 85_H to C register.</p> <p>REGISTERS</p> <table border="1"> <tr> <td>A</td> <td>XX</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>85</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	A	XX	XX	F	B	XX	85	C	D	XX	XX	E	H	XX	XX	L
A	XX	XX	F															
B	XX	85	C															
D	XX	XX	E															
H	XX	XX	L															
AGAIN	DCR C	<p>Decrement the C register content one by one in each step.</p>																
	JNZ AGAIN	<p>Jump on no Zero(Z=0). If C register content is not equal to zero, then go on looping. If C register content is equal to zero then come out of the loop(AGAIN) & continue sequentially.</p> <p>REGISTERS</p> <table border="1"> <tr> <td>A</td> <td>XX</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>00</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	A	XX	XX	F	B	XX	00	C	D	XX	XX	E	H	XX	XX	L
A	XX	XX	F															
B	XX	00	C															
D	XX	XX	E															
H	XX	XX	L															
	RET	<p>The program sequence is transferred from subroutine to the calling program.</p>																

TRIANGULAR WAVEFORM GENERATION

ALGORITTHM

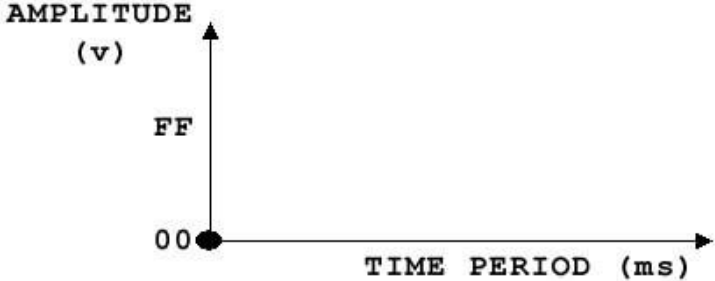
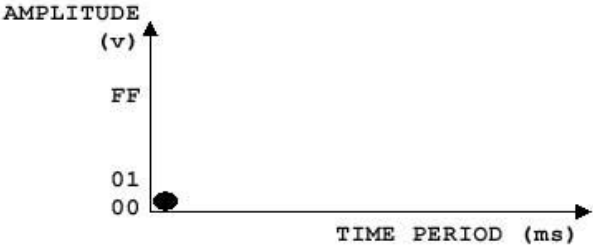
- 1.Intialization a control word for 8255,for it to operate in I/O mode and for ports A, B and C to operate in output mode.
- 2.Clear the accumulator content and output it.
- 3.Increment accumulator content and compare with FF_H
- 4.Jump if nor zero to step 2.
- 5.Decrement accumulator content.
- 6.Output it and compare with 00_H and go to step5 if not zero.
- 7.Continue the above steps.

ASSEMBLY LANGUAGE PROGRAM

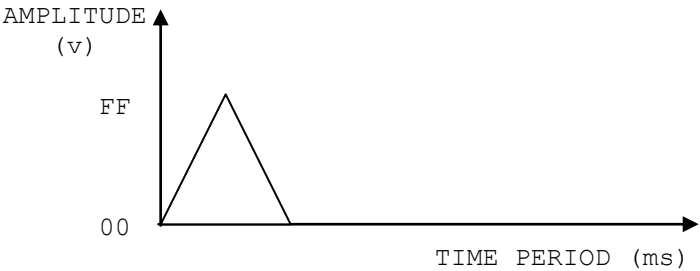
ADDRESS	LABEL	MNEMONICS	OPCODE/OPERAND
C500		MVI A, 80 _H	3E 80
C502		OUT CWR	D3 DB
C504	START	MVI A, 00 _H	3E 00
C506	POS	OUT PORTA	D3 D8
C508		INR A	3C
C509		CPI FF _H	FE FF
C50B		JNZ POS	C2 06 C5
C50E	NEG	DCR A	3D
C50F		OUT PORTA	D3 D8
C511		CPI 00 _H	FE 00
C513		JNZ NEG	C2 0E C5
C516		JMP START	C3 04 C5

PROGRAM TRACE

LABEL	MNEMONICS	DESCRIPTION																																											
	MVI A, 80 _H	<p>Initializing the ports of the PPI 8255 as O/P ports by writing the control word as 80_H.</p> <table border="1" style="width: 100%; border-collapse: collapse; margin: 10px 0;"> <thead> <tr> <th style="text-align: left;">DATA BITS</th> <th style="text-align: center;">D₇</th> <th style="text-align: center;">D₆</th> <th style="text-align: center;">D₅</th> <th style="text-align: center;">D₄</th> <th style="text-align: center;">D₃</th> <th style="text-align: center;">D₂</th> <th style="text-align: center;">D₁</th> <th style="text-align: center;">D₀</th> </tr> </thead> <tbody> <tr> <td></td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> <tr> <td style="text-align: left;">COMMENT</td> <td style="text-align: center;">I/O mode</td> <td style="text-align: center;">Mode0</td> <td style="text-align: center;">PortA O/P</td> <td style="text-align: center;">PortC Upper O/P</td> <td style="text-align: center;">Mode0</td> <td style="text-align: center;">PortB O/P</td> <td style="text-align: center;">PortC Lower O/P</td> <td></td> </tr> </tbody> </table> <p>80_H is moved to accumulator.</p> <p style="text-align: center;">REGISTERS</p> <table style="margin-left: 20px;"> <tr> <td style="padding-right: 5px;">A</td> <td style="border: 1px solid black; padding: 2px 5px;">80</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="padding-left: 5px;">F</td> </tr> <tr> <td>B</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td>C</td> </tr> <tr> <td>D</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td>E</td> </tr> <tr> <td>H</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td style="border: 1px solid black; padding: 2px 5px;">XX</td> <td>L</td> </tr> </table>	DATA BITS	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		1	0	0	0	0	0	0	0	COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P		A	80	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
DATA BITS	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																					
	1	0	0	0	0	0	0	0																																					
COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P																																						
A	80	XX	F																																										
B	XX	XX	C																																										
D	XX	XX	E																																										
H	XX	XX	L																																										
	OUT CWR	Control word specify the I/O function for each ports of 8255.																																											

START	MVI A,00 _H	<p>Initialize portA as 00_H .</p> <p>REGISTERS</p> <table border="1" data-bbox="560 293 775 421"> <tr> <td>A</td> <td>00</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>XX</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	A	00	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
A	00	XX	F															
B	XX	XX	C															
D	XX	XX	E															
H	XX	XX	L															
POS	OUT PORTA	<p>00_H is outputted through portA. This is the base point of the triangular waveform.</p> 																
	INR A	<p>Increment the accumulator content. This helps in generating the positive slope of the triangular waveform.</p> <p>REGISTERS</p> <table border="1" data-bbox="560 1066 775 1193"> <tr> <td>A</td> <td>01</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>XX</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table> 	A	01	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
A	01	XX	F															
B	XX	XX	C															
D	XX	XX	E															
H	XX	XX	L															
	CPI FF _H	<p><i>Compare Immediate</i> The content of the second byte of the instruction is subtracted from the accumulator. The condition flags are set by the result of the subtraction. The Z flag is set to 1 if (A)=(byte2). The CY flag is set to 1 if (A)<(byte2). If accumulator content equal to FF_H then Z flag is set, otherwise Z flag is reset.</p>																
	JNZ POS	<p>Jump on no Zero(Z=0). No flags are affected. If accumulator content is not equal to FF_H, then the control is transferred to the instruction whose label is POS. If accumulator content is equal to FF_H continues sequentially.</p>																

		<p>REGISTERS</p> <table border="1"> <tr><td>A</td><td>FF</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr> </table> <p>FLAG WORD</p> <table border="1"> <tr><td>S</td><td>Z</td><td>-</td><td>AC</td><td>-</td><td>P</td><td>-</td><td>C</td></tr> <tr><td>X</td><td>1</td><td></td><td>X</td><td></td><td>X</td><td></td><td>X</td></tr> </table> <p>The FF_H is the peak point of the triangular waveform.</p> <p>AMPLITUDE (v)</p> <p>00</p> <p>FF</p> <p>TIME PERIOD (ms)</p>	A	FF	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L	S	Z	-	AC	-	P	-	C	X	1		X		X		X
A	FF	XX	F																															
B	XX	XX	C																															
D	XX	XX	E																															
H	XX	XX	L																															
S	Z	-	AC	-	P	-	C																											
X	1		X		X		X																											
NEG	DCR A	<p>Decrement the accumulator content. FE_H is in the accumulator. This helps in generating the negative slope of the triangular waveform .</p> <p>REGISTERS</p> <table border="1"> <tr><td>A</td><td>FE</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr> </table>	A	FE	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L																
A	FE	XX	F																															
B	XX	XX	C																															
D	XX	XX	E																															
H	XX	XX	L																															
	OUT PORTA	<p>FE_H is outputted through portA.</p> <p>AMPLITUDE (v)</p> <p>00</p> <p>FE</p> <p>FF</p> <p>TIME PERIOD (ms)</p>																																
	CPI 00 _H	<p>If accumulator content equal to 00_H then Z flag is set, otherwise Z flag is reset.</p>																																
	JNZ NEG	<p>If accumulator content is not equal to 00_H, then the control is transferred to the instruction whose label is NEG. If accumulator content is equal to 00_H continues sequentially.</p> <p>REGISTERS</p> <table border="1"> <tr><td>A</td><td>00</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr> </table> <p>FLAG WORD</p>	A	00	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L																
A	00	XX	F																															
B	XX	XX	C																															
D	XX	XX	E																															
H	XX	XX	L																															

		<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>S</td><td>Z</td><td>-</td><td>AC</td><td>-</td><td>P</td><td>-</td><td>C</td> </tr> <tr> <td>X</td><td>1</td><td></td><td>X</td><td></td><td>X</td><td></td><td>X</td> </tr> </table> <p>The 00_H is the end point of the triangular waveform.</p> 	S	Z	-	AC	-	P	-	C	X	1		X		X		X
S	Z	-	AC	-	P	-	C											
X	1		X		X		X											
	JMP START	The triangular waveform was generated continuously by repeating the loop(START).																

SAWTOOTH WAVEFORM GENERATION

ALGORITHM

1. Initialization a control word for 8255, for it to operate in I/O mode and for ports A,B and C to operate in output mode.
2. Clear the accumulator content and output it.
3. Increment accumulator content and compare with FF_H.
4. Jump if not zero to step 2.
5. Continue the above steps.

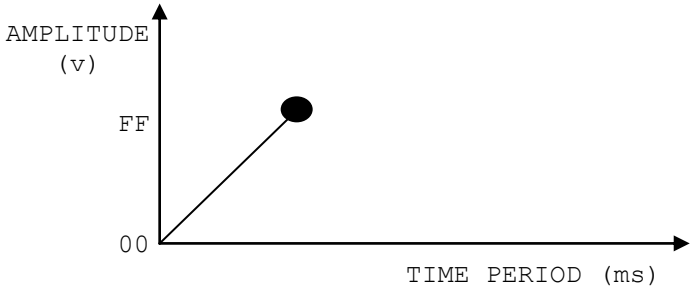
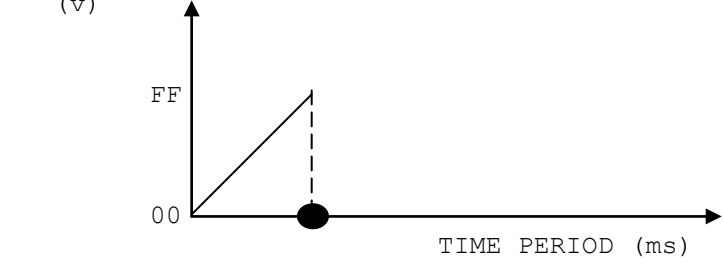
ASSEMBLY LANGUAGE PROGRAM

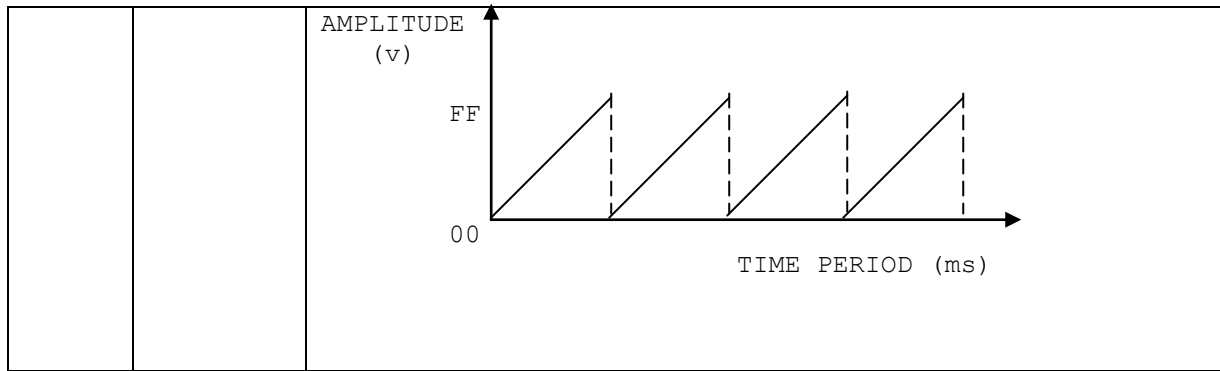
ADDRESS	LABEL	MNEMONICS	OPCODE/OPERAND
C600		MVI A, 80 _H	3E 80
C602		OUT CWR	D3 DB
C604	START	MVI A, 00 _H	3E 00
C606	REPEAT	OUT PORTA	D3 D8
C608		INR A	3C
C609		CPI FF _H	FE FF
C60B		JNZ REPEAT	C2 06 C6
C60E		MVI A, 00 _H	3E 00
C610		OUT PORTA	D3 D8
C612		JMP START	C3 04 C6

PROGRAM TRACE

LABEL	MNEMONICS	DESCRIPTION									
	MVI A, 80 _H	Initializing the ports of the PPI 8255 as O/P ports by writing the control word as 80 _H .									
		<table border="1" style="width: 100%; text-align: center;"> <tr> <td>DATA</td> <td>D₇</td> <td>D₆</td> <td>D₅</td> <td>D₄</td> <td>D₃</td> <td>D₂</td> <td>D₁</td> <td>D₀</td> </tr> </table>	DATA	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
DATA	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀			

		<table border="1"> <tr> <td>BITS</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>COMMENT</td> <td>I/O mode</td> <td>Mode0</td> <td>PortA O/P</td> <td>PortC Upper O/P</td> <td>Mode0</td> <td>PortB O/P</td> <td>PortC Lower O/P</td> <td></td> </tr> </table> <p>80_H is moved to accumulator.</p> <p>REGISTERS</p> <table border="1"> <tr> <td>A</td> <td>80</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>XX</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	BITS	1	0	0	0	0	0	0	0	COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P		A	80	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
BITS	1	0	0	0	0	0	0	0																												
COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P																													
A	80	XX	F																																	
B	XX	XX	C																																	
D	XX	XX	E																																	
H	XX	XX	L																																	
	OUT CWR	CWR specify the I/O function for each ports of 8255.																																		
START	MVI A,00 _H	<p>Initialize portA as 00_H .</p> <p>REGISTERS</p> <table border="1"> <tr> <td>A</td> <td>00</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>XX</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	A	00	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L																		
A	00	XX	F																																	
B	XX	XX	C																																	
D	XX	XX	E																																	
H	XX	XX	L																																	
REPEAT	OUT PORTA	<p>00_H is outputted through portA. This is the base point of the sawtooth waveform.</p>																																		
	INR A	<p>Increment the accumulator content. This helps in generating the slope of the sawtooth waveform.</p> <p>REGISTERS</p> <table border="1"> <tr> <td>A</td> <td>01</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>XX</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>XX</td> <td>XX</td> <td>L</td> </tr> </table>	A	01	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L																		
A	01	XX	F																																	
B	XX	XX	C																																	
D	XX	XX	E																																	
H	XX	XX	L																																	
	CPI FF _H	<p>Compare Immediately the accumulator content with FF_H. If accumulator content equal to FF_H then Z flag is set, otherwise Z flag is reset.</p>																																		

<p>JNZ REPEAT</p>	<p>Jump on no Zero(Z=0). No flags are affected. If accumulator content is not equal to FF_H, then the control is transferred to the instruction whose label is REPEAT. If accumulator content is equal to FF_H continues sequentially.</p>	<p>REGISTERS</p> <table border="1" style="display: inline-table; margin-right: 10px;"> <tr><td>A</td><td>FF</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr> </table> <p>FLAG WORD</p> <table border="1" style="display: inline-table; margin-right: 10px;"> <tr><td>S</td><td>Z</td><td>-</td><td>AC</td><td>-</td><td>P</td><td>-</td><td>C</td></tr> <tr><td>X</td><td>1</td><td></td><td>X</td><td></td><td>X</td><td></td><td>X</td></tr> </table> <p>The FF_H is the peak point of the sawtooth waveform.</p> 	A	FF	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L	S	Z	-	AC	-	P	-	C	X	1		X		X		X
A	FF	XX	F																															
B	XX	XX	C																															
D	XX	XX	E																															
H	XX	XX	L																															
S	Z	-	AC	-	P	-	C																											
X	1		X		X		X																											
<p>MVI A,00_H</p>	<p>The 00_H is the base point of the sawtooth waveform. The pointer transfers to the base point.</p>	<p>REGISTERS</p> <table border="1" style="display: inline-table; margin-right: 10px;"> <tr><td>A</td><td>00</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr> </table>	A	00	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L																
A	00	XX	F																															
B	XX	XX	C																															
D	XX	XX	E																															
H	XX	XX	L																															
<p>OUT PORTA</p>	<p>00_H is outputted through portA.</p>	<p>AMPLITUDE (v)</p> 																																
<p>JMP START</p>	<p>The sawtooth waveform was generated continuously by repeating the loop(START).</p>																																	



SINE WAVEFORM GENERATION

ALGORITHM

1. Initialization a control word for 8255, for it to operate in I/O mode and for ports A, B and C to operate in output mode.
2. Initialize the HL register pair.
3. Move memory content to accumulator and output it.
4. Increment the HL register pair content. Decrement C register content.
5. Jump if no zero to step3 and move 24 into C register .
6. Decrement the content of HL register pair.
7. Move memory content to accumulator and output it.
8. Decrement it and compare with zero. If no zero jump to step6.
9. Continue the above steps.

ASSEMBLY LANGUAGE PROGRAM

ADDRESS	LABEL	MNEMONICS	OPCODE/OPERAND
C300		MVI A, 80 _H	3E 80
C302		OUT CWR	D3 DB
C304	START	MVI C, 24 _H	0E 24
C306		LXI H, C900 _H	21 00 C4
C309	POS	MOV A, M	7E
C30A		OUT PORTA	D3 D8
C30C		INX H	23
C30D		DCR C	0D
C30E		JNZ POS	C2 09 C3
C311		MVI C, 24 _H	0E 24
C313	NEG	DCX H	2B
C314		MOV A, M	7E
C315		OUT PORTA	D3 D8
C317		DCR C	0D
C318		JNZ NEG	C2 13 C3
C31B		JMP START	C3 04 C3

PROGRAM TRACE

LABEL	MNEMONICS	DESCRIPTION																																											
	MVI A,80 _H	<p>Initializing the ports of the PPI 8255 as O/P ports by writing the control word as 80_H.</p> <table border="1" data-bbox="422 398 1447 555"> <tr> <td>DATA BITS</td> <td>D₇</td> <td>D₆</td> <td>D₅</td> <td>D₄</td> <td>D₃</td> <td>D₂</td> <td>D₁</td> <td>D₀</td> </tr> <tr> <td></td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>COMMENT</td> <td>I/O mode</td> <td>Mode0</td> <td>PortA O/P</td> <td>PortC Upper O/P</td> <td>Mode0</td> <td>PortB O/P</td> <td>PortC Lower O/P</td> <td></td> </tr> </table> <p>80_H is moved to accumulator.</p> <p>REGISTERS</p> <table border="1" data-bbox="483 645 699 768"> <tr><td>A</td><td>80</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr> </table>	DATA BITS	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀		1	0	0	0	0	0	0	0	COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P		A	80	XX	F	B	XX	XX	C	D	XX	XX	E	H	XX	XX	L
DATA BITS	D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀																																					
	1	0	0	0	0	0	0	0																																					
COMMENT	I/O mode	Mode0	PortA O/P	PortC Upper O/P	Mode0	PortB O/P	PortC Lower O/P																																						
A	80	XX	F																																										
B	XX	XX	C																																										
D	XX	XX	E																																										
H	XX	XX	L																																										
	OUT CWR	Control word specify the I/O function for each ports of 8255.																																											
START	MVI C,24 _H	<p>Initialize C register as 24_H.</p> <p>REGISTERS</p> <table border="1" data-bbox="483 880 699 1003"> <tr><td>A</td><td>80</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>24</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr> </table> <p>Setting count for look up table data sequence i.e. 24 data are loaded from memory for sine waveform generation in <i>positive direction</i>.</p>	A	80	XX	F	B	XX	24	C	D	XX	XX	E	H	XX	XX	L																											
A	80	XX	F																																										
B	XX	24	C																																										
D	XX	XX	E																																										
H	XX	XX	L																																										
	LXI H,C900 _H	<p>Initialize the memory pointer at C900_H .i.e. loads the 16-bit data in the register pair designated.</p> <p>REGISTERS</p> <table border="1" data-bbox="483 1216 699 1339"> <tr><td>A</td><td>80</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>24</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>00</td><td>L</td></tr> </table> <p>C900_H is the memory pointer to the starting address of the sine waveform look up table data sequence.</p> <p>MEMORY</p> <table border="1" data-bbox="483 1451 1042 1619"> <tr><td>C900</td><td>00</td><td>← HL memory pointer</td></tr> <tr><td>C901</td><td>01</td><td></td></tr> <tr><td>C902</td><td>02</td><td></td></tr> <tr><td>C903</td><td>04</td><td></td></tr> <tr><td>C904</td><td>08</td><td></td></tr> </table>	A	80	XX	F	B	XX	24	C	D	XX	XX	E	H	C9	00	L	C900	00	← HL memory pointer	C901	01		C902	02		C903	04		C904	08													
A	80	XX	F																																										
B	XX	24	C																																										
D	XX	XX	E																																										
H	C9	00	L																																										
C900	00	← HL memory pointer																																											
C901	01																																												
C902	02																																												
C903	04																																												
C904	08																																												
POS	MOV A,M	<p>Memory pointer content 00_H is moved to accumulator.</p> <p>REGISTERS</p> <table border="1" data-bbox="483 1697 699 1821"> <tr><td>A</td><td>00</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>24</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>00</td><td>L</td></tr> </table>	A	00	XX	F	B	XX	24	C	D	XX	XX	E	H	C9	00	L																											
A	00	XX	F																																										
B	XX	24	C																																										
D	XX	XX	E																																										
H	C9	00	L																																										
	OUT PORTA	<p>00_H is outputted through portA. This is the beginning of the sine waveform in positive direction .i.e.Storing the accumulator content in the DAC data latches.</p>																																											

<p>INX H</p>	<p>Increment the HL register pair by 1. The instruction views the contents of the HL registers as a 16-bit number. No flags are affected.</p>	<p>REGISTERS</p> <table border="1" style="display: inline-table; margin-right: 20px;"> <tr><td>A</td><td>00</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>24</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>01</td><td>L</td></tr> </table> <p>C901_H is the memory pointer of the sine waveform look up table data sequence.</p> <p>MEMORY</p> <table border="1" style="display: inline-table; margin-right: 20px;"> <tr><td>C900</td><td>00</td></tr> <tr><td>C901</td><td>01</td></tr> <tr><td>C902</td><td>02</td></tr> <tr><td>C903</td><td>04</td></tr> <tr><td>C904</td><td>08</td></tr> </table> <div style="border: 1px solid black; padding: 2px; display: inline-block;">HL memory pointer</div>	A	00	XX	F	B	XX	24	C	D	XX	XX	E	H	C9	01	L	C900	00	C901	01	C902	02	C903	04	C904	08
A	00	XX	F																									
B	XX	24	C																									
D	XX	XX	E																									
H	C9	01	L																									
C900	00																											
C901	01																											
C902	02																											
C903	04																											
C904	08																											
<p>DCR C</p>	<p>Decrement the C register content one by one in each step.</p>	<p>REGISTERS</p> <table border="1" style="display: inline-table; margin-right: 20px;"> <tr><td>A</td><td>00</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>23</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>01</td><td>L</td></tr> </table>	A	00	XX	F	B	XX	23	C	D	XX	XX	E	H	C9	01	L										
A	00	XX	F																									
B	XX	23	C																									
D	XX	XX	E																									
H	C9	01	L																									
<p>JNZ POS</p>	<p>Jump on no Zero (Z=0). If C register content is not equal to zero, then go on looping. If C register content is equal to zero then come out of the loop (POS) & continues sequentially.</p>	<p>REGISTERS</p> <table border="1" style="display: inline-table; margin-right: 20px;"> <tr><td>A</td><td>FF</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>00</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>23</td><td>L</td></tr> </table>	A	FF	XX	F	B	XX	00	C	D	XX	XX	E	H	C9	23	L										
A	FF	XX	F																									
B	XX	00	C																									
D	XX	XX	E																									
H	C9	23	L																									
<p>MVI C, 24_H</p>	<p>Reinitialize C register as 24_H.</p>																											

		<p>REGISTERS</p> <table border="1"> <tr><td>A</td><td>FF</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>24</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>23</td><td>L</td></tr> </table> <p>Setting count for look up table data sequence i.e. 24 data are loaded from memory for sine waveform generation in <i>negative direction</i>.</p>	A	FF	XX	F	B	XX	24	C	D	XX	XX	E	H	C9	23	L										
A	FF	XX	F																									
B	XX	24	C																									
D	XX	XX	E																									
H	C9	23	L																									
NEG	DCX H	<p>Decrement the HL register pair by 1. The instruction views the contents of the HL registers as a 16-bit number. No flags are affected.</p> <p>REGISTERS</p> <table border="1"> <tr><td>A</td><td>FF</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>24</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>22</td><td>L</td></tr> </table> <p>C922_H is the memory pointer of the sine waveform look up table data sequence.</p> <p>MEMORY</p> <table border="1"> <tr><td>C91F</td><td>F2</td></tr> <tr><td>C920</td><td>F9</td></tr> <tr><td>C921</td><td>FC</td></tr> <tr><td>C922</td><td>FD</td></tr> <tr><td>C923</td><td>FF</td></tr> </table> <p style="text-align: right;">HL memory pointer</p>	A	FF	XX	F	B	XX	24	C	D	XX	XX	E	H	C9	22	L	C91F	F2	C920	F9	C921	FC	C922	FD	C923	FF
A	FF	XX	F																									
B	XX	24	C																									
D	XX	XX	E																									
H	C9	22	L																									
C91F	F2																											
C920	F9																											
C921	FC																											
C922	FD																											
C923	FF																											
	MOV A,M	<p>Memory pointer content FD_H is moved to accumulator.</p> <p>REGISTERS</p> <table border="1"> <tr><td>A</td><td>FD</td><td>XX</td><td>F</td></tr> <tr><td>B</td><td>XX</td><td>24</td><td>C</td></tr> <tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr> <tr><td>H</td><td>C9</td><td>22</td><td>L</td></tr> </table>	A	FD	XX	F	B	XX	24	C	D	XX	XX	E	H	C9	22	L										
A	FD	XX	F																									
B	XX	24	C																									
D	XX	XX	E																									
H	C9	22	L																									
	OUT PORTA	<p>FD_H is outputted through portA. This is the beginning of the sine waveform in negative direction.</p>																										
	DCR C	Decrement the C register content one by one in each step.																										

		<p align="center">REGISTERS</p> <table border="1"> <tr> <td>A</td> <td>FE</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>23</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>C9</td> <td>22</td> <td>L</td> </tr> </table>	A	FE	XX	F	B	XX	23	C	D	XX	XX	E	H	C9	22	L
A	FE	XX	F															
B	XX	23	C															
D	XX	XX	E															
H	C9	22	L															
	JNZ NEG	<p>Jump on no Zero(Z=0). If C register content is not equal to zero, then go on looping. If C register content is equal to zero then come out of the loop(NEG) & continues sequentially.</p> <p align="center">REGISTERS</p> <table border="1"> <tr> <td>A</td> <td>00</td> <td>XX</td> <td>F</td> </tr> <tr> <td>B</td> <td>XX</td> <td>00</td> <td>C</td> </tr> <tr> <td>D</td> <td>XX</td> <td>XX</td> <td>E</td> </tr> <tr> <td>H</td> <td>C9</td> <td>00</td> <td>L</td> </tr> </table>	A	00	XX	F	B	XX	00	C	D	XX	XX	E	H	C9	00	L
A	00	XX	F															
B	XX	00	C															
D	XX	XX	E															
H	C9	00	L															
	JMP START	<p>The sine waveform was generated continuously by repeating the cycle.</p>																

LOOK UP TABLE

$$\text{ALPHA} = 128 + 128 \text{ SIN} (\text{THETA})$$

Where THETA = -90 degree to +90 degree for every 5 degree.

ADDRESS	ANGLE	DEGREE	HEX CODE	DATA (HEX CODE Approximate)
C900	-90	0.0	00	00
C901	-85	0.48	00	01
C902	-80	1.94	01	02
C903	-75	4.36	04	04
C904	-70	7.719	08	08
C905	-65	11.99	0B	11
C906	-60	17.14	11	17
C907	-55	23.14	17	1E
C908	-50	29.94	1D	25
C909	-45	37.49	25	2D
C90A	-40	45.72	2E	36
C90B	-35	54.58	37	40
C90C	-30	64.00	40	49
C90D	-25	73.90	4A	54

C90E	-20	84.22	54	5E
C90F	-15	94.87	5F	69
C910	-10	105.77	6A	74
C911	-05	116.84	74	7F
C912	00	128.00	80	84
C913	05	139.15	8B	95
C914	10	150.20	96	9F
C915	15	161.12	A1	AF
C916	20	171.77	A6	B4
C917	25	182.09	B6	C0
C918	30	192.00	C0	C8
C919	35	201.42	C9	D0
C91A	40	210.27	D2	D8
C91B	45	218.50	DA	E0
C91C	50	226.05	E2	EA
C91D	55	232.85	E8	ED
C91E	60	238.85	EE	EF
C91F	65	244.00	F4	F2
C920	70	248.28	F8	F9
C921	75	251.63	FB	FC
C922	80	254.05	FE	FD
C923	85	255.51	FF	FF
C924	90	256.00	100	00

TIPS

IC	ADDRESS			
	PORT A	PORT B	PORT C	CONTROL
<i>PPI 8255 (U4)</i>	D8	D9	DA	DB
<i>PPI 8255 (U3)</i>	F0	F1	F2	F3

OBSERVATION

WAVEFORM	AMPLITUDE (V)	TIME PERIOD (ms)	FREQUENCY (Hz)
SQUARE	0.001	0.6	1.67
TRIANGLE	4.4	1.3	0.77
SAWTOOTH	1.1	2.6	0.38
SINE	1.0	0.88	1.14

VERIFICATION

Check the waveform at the output lines of the 8255. The frequency of the waveform depends solely on the time delay routine used.

REFERENCE

1. Ramesh S.Gaonkar, Microprocessor Architecture, Programming, and Applications, Fourth Edition, Penram International Publishing (India), 2000.

2. S.Subathra, Advanced Microprocessor Lab, Record work, Adhiparashakthi Engineering College, Melmaruvathur, October 2002
3. S.Subathra, "Programming in 8085 Microprocessor and its applications – An Innovative Analysis", Technical Report, Adhiparashakthi Engineering College, Melmaruvathur, March 2003
4. Micro-85 EB, User Manual, Version – 3.0, CAT #M85 EB-002, VI Microsystems Pvt. Ltd., Chennai.