TI DSP, MCU, Xilinx Zynq FPGA 기반의 프로그래밍 전문가 과정

<디지털 신호처리>

2018.06.11 – 71 일차

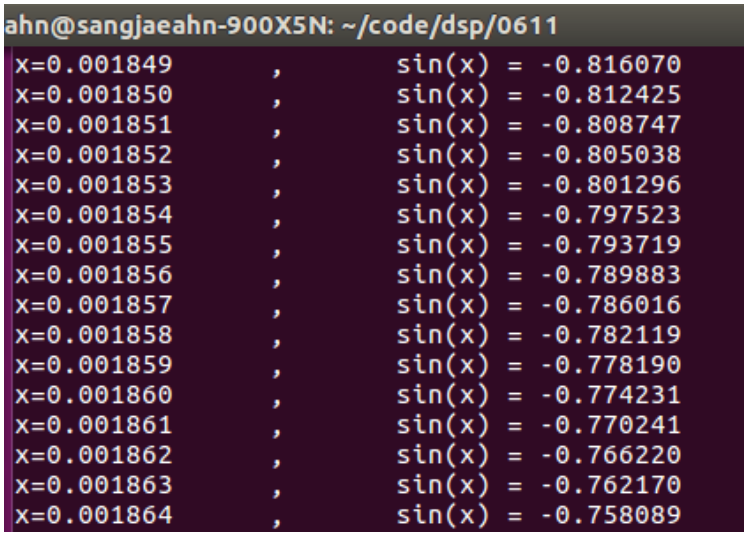
강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com

학생 - 안상재
sangjae2015@naver.com

1. sin(wx) 함수의 이산값 구하기!
- sin(wx) = sin(2*pi*f*x)
- 샘플링주기 : 0.000001 , 주파수(f) : 1khz
- sin(wx) 의 주기 : 0.001
- 0.002 (2 주기 구간)까지의 이산값을 계산하고 출력함

```
 1 #include <stdio.h>
 2 #include <math.h>
 3
 4 #define pi 3.14
 5 #define sample 0.000001
 6 #define f 1000      // f = 1khz
 7
 8 float func(float x)
 9 {
10     return sin(2*pi*f*x);
11 }
12
13 int main(void)
14 {
15     float i;
16     float x;
17
18     for(i=0;i<=0.002;i+=sample)
19     {
20         printf("x=%f    ,   sin(x) = %f\n", i, func(i));
21     }
22     return 0;
23 }
```

1-1. 결과 분석



<opengl 설치하기>

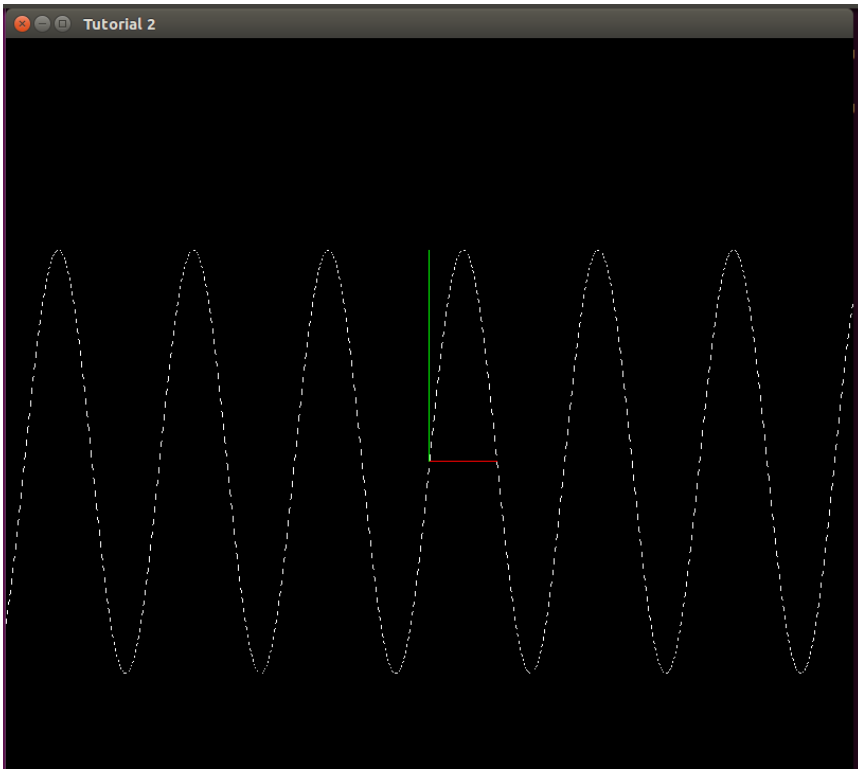
1. sudo apt-get update
2. sudo apt-get install build-essential
3. sudo apt-get install freeglut3 freeglut3-dev
4. sudo apt-get install glew-utills glee-dev
5. sudo apt-get install libglew-dev

2. sin 파형 출력하기

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
#include <GL/glu.h>
#include <GL/gl.h>
#include <GL/freeglut.h>

#define pi 3.14

void originAxis(void);
void sineWave(void);
void idle(void);
void rect(void);

void display(void)
{
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

	originAxis();
	sineWave();
	glutSwapBuffers();
}

void sineWave(void)
{
	float wavelength = 2;
	float amplitude = 1;
	float sample = 0.01;
	float k, x, y;
	glBegin(GL_LINES);
	glColor3f(1,1,1);

	for(x=-10*pi;x<=10*pi;x+=sample){
		k = 2 * 3.14 / wavelength;
		y = amplitude * sin(k * x);
		glVertex3f(x, y, 0);
	}

	glEnd();
}

void originAxis(void)		// x,y,z 축 그리기
{
	glBegin(GL_LINES);
```

```
        glColor3f(1,0,0);
        glVertex3f(0,0,0);
        glVertex3f(1,0,0);
        glColor3f(0,1,0);
        glVertex3f(0,0,0);
        glVertex3f(0,1,0);
        glColor3f(0,0,1);
        glVertex3f(0,0,0);
        glVertex3f(0,0,1);
        glEnd();
}

int main(int argc, char **argv)
{
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
        glutInitWindowSize(800, 800);
        glutCreateWindow("Tutorial 2");

        glOrtho(-2*pi, 2*pi, -2, 2, -1, 1);
        glEnable(GL_DEPTH_TEST);

        glutDisplayFunc(display);  // 화면에 표시하기
        glutIdleFunc(idle);
        glutMainLoop();

        return EXIT_SUCCESS;
}
```

2-1. 결과 분석



3. opengl 기반으로 사각파(Rectangular Wave) 출력하기!

- 사각파를 푸리에 급수의 수식으로 표현함

(푸리에 급수의 항을 최대한 많이 더할 수록 정확한 파형을 얻을 수 있으나 , 연산량이 많아져서 출력 시간이 올래 걸림.)

- 파형, x 축, y 축의 색을 표현함

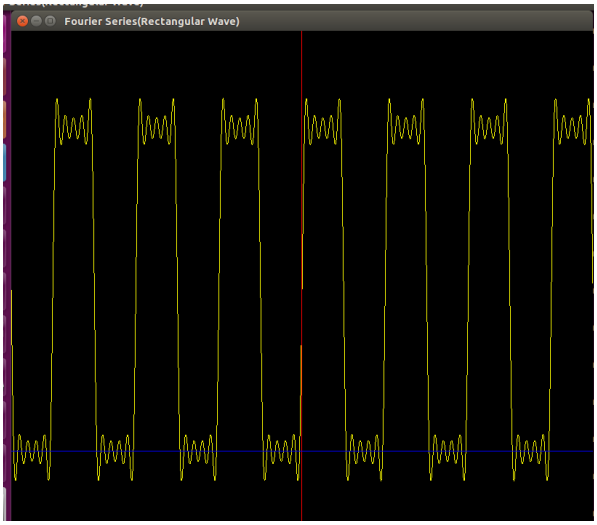
- 컴파일 방법 : gcc scale_rect_wave.c -lGL -lglut -lGLU -lm

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
#include <GL/glu.h>
#include <GL/gl.h>
#include <GL/freeglut.h>

void originAxis(void);
void sineWave(void);
void idle(void);

void display(void)
{
	glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

	originAxis();
	sineWave();

	glutSwapBuffers();
}

void sineWave(void)
{
	float wavelength = 2.0 * M_PI;
	float amplitude = 1;
	float inc = 2.0 * M_PI / 1024.0;
	float k, x, y, yp = 0, y2, y2p = 0, cx, cy, cy2;
	int i, cache = 0;

	glBegin(GL_LINES);
	glColor3f(1,1,0);
	for(x=-7*M_PI;x<=7*M_PI;x+=inc)
	{
		yp = 0;

		for(i = 1; i < 10000; i++)	/* 퓨리에 급수의 sin 항을 얼마나 더할 것인가(많이 더할수록
                                                정확한 파형이 생성됨 ) */
			yp += ((1.0 - cos(i * M_PI)) / (i * M_PI)) * sin(i * x); // 사각파의 푸리에 급수 표현

		y = yp + 0.5;

		if(cache)
		{
			glVertex2f(cx, cy);
			glVertex2f(x, y);
```

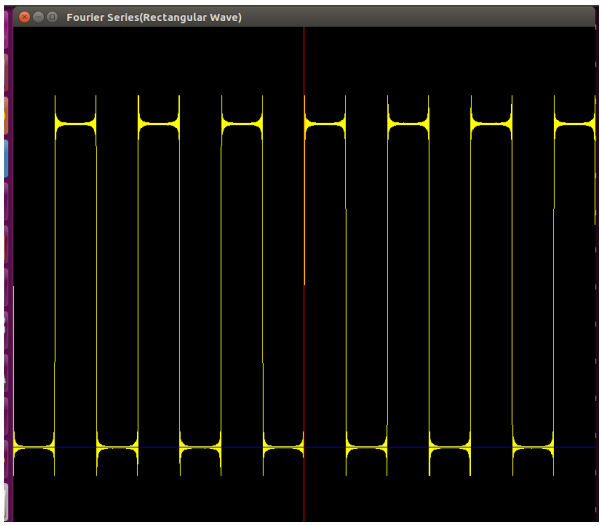
```
                }

                cache = 1;
                cx = x;
                cy = y;
        }
        glEnd();

        cache = 0;
}

void originAxis(void)
{
        glBegin(GL_LINES);
        glColor3f(0,0,1);
        glVertex3f(-100,0,0);
        glVertex3f(100, 0, 0);
        glColor3f(1,0,0);
        glVertex3f(0,-100,0);
        glVertex3f(0, 100, 0);
        glColor3f(0,0,1);
        glVertex3f(0,0,0);
        glVertex3f(0, 0, 1);
        glEnd();
}

int main(int argc, char **argv)
{
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH);
        glutInitWindowSize(800, 800);     // 윈도우의 크기
        glutCreateWindow("Fourier Series(Rectangular Wave)");     // 윈도우 이름

        glOrtho(-7 * M_PI, 7 * M_PI, -0.5, 1.3, -1.0, 1.0);       // 표시한 그래프를 확대, 축소함
        glEnable(GL_DEPTH_TEST);

        glutDisplayFunc(display);  // 그래프 출력
        glutMainLoop();      // 윈도우 창 출력

        return EXIT_SUCCESS;
}
```

3-1. 결과 분석

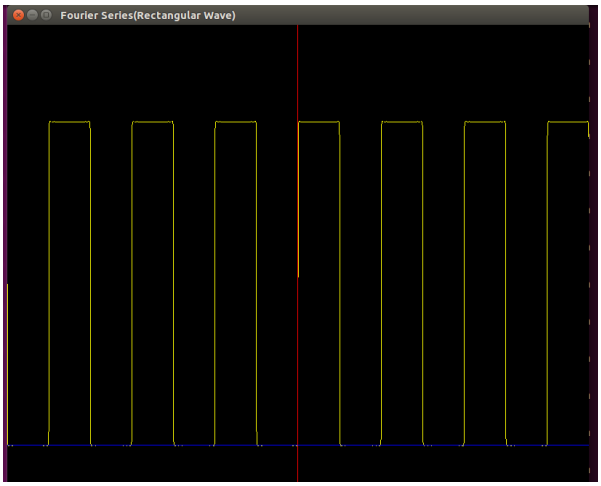
1) 푸리에 급수항 10 개 합한 결과



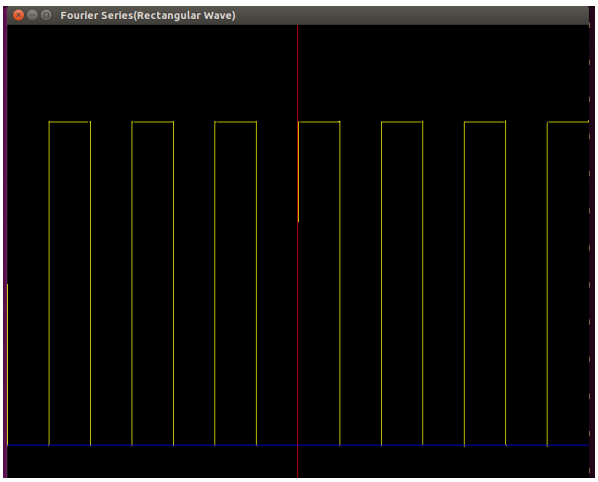
2) 푸리에 급수항 100 개 합한 결과



3) 푸리에 급수항 1000 개 합한 결과



4) 푸리에 급수항 10000 개 합한 결과



→ 푸리에 급수 항을 많이 합할 수록 정확한 파형을 얻을 수 있다.