TI DSP, MCU, Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee (이상훈)
gcccompil3r@gmail.com
학생 – 김형주
mihaelkel@naver.com

vector.h

```c
#ifndef __VECTOR_H__
#define __VECTOR_H__
#include <stdio.h>
#include <math.h>
void print_vector(double* x);
void vector_sum(double* res, double* x, double* y);
void vector_sub(double* res, double* x, double* y);
void vector_scalar_mul(double* res, double* x, double scalar);
double vector_inner_product(double* x, double* y);
void vector_cross_product(double* res, double* x, double* y);
double vector_abs_val(double* x);
void vector_gso_normalization(double* x0, double* x1, double* x2,double* w0,double* w1,double* w2);
#endif
```

vector.c

```c
#include "vector.h"
#include <stdio.h>
#include <math.h>
void print_vector(double* x)
{
    printf("%lf, %lf, %lf\n",x[0],x[1],x[2]);
}
void vector_sum(double* res,double* x, double* y){
    int i;
    for(i=0;i<3;i++)
        res[i] = x[i] + y[i];
}
void vector_sub(double* res,double* x, double* y){
    int i;
    for(i=0;i<3;i++)
        res[i] = x[i] - y[i];
}
void vector_scalar_mul(double* res, double* x, double scalar)
{
    int i;
    for(i=0;i<3;i++)
        res[i] = x[i]*scalar;
}
double vector_inner_product(double* x, double* y)
{
    int i;
    double res = 0;
    for(i=0;i<3;i++)
        res += x[i]*y[i];
    return res;
}
void vector_cross_product(double* res, double* x, double* y)
{
    int i;
    for(i=0;i<3;i++)
        res[i] = x[(i+1)%3]*y[(i+2)%3]-x[(i+2)%3]*y[(i+1)%3];

}
double vector_abs_val(double* x)
```

```c
{
    int i;
    double res = 0;
    for(i=0;i<3;i++)
        res += x[i]*x[i];
    res = sqrt(res);
    return res;
}
void vector_gso_normalization(double* x0, double* x1, double* x2,double* w0, double* w1, double* w2)
{
    int i;
    for(i=0;i<3;i++)
        w0[i] = x0[i];

    for(i=0;i<3;i++)
    {
        w1[i] = x1[i] - ((vector_inner_product(x1,w0))/pow(vector_abs_val(w0),2))*w0[i];
    }

    for(i=0;i<3;i++)
    {
        w2[i] = x2[i] - ((vector_inner_product(x2,w0))/pow(vector_abs_val(w0),2))*w0[i]
                    - ((vector_inner_product(x2,w1))/pow(vector_abs_val(w1),2))*w1[i];
    }


}
```

```c
#include "vector.h"
int main(void)
{
    double v0[3] = {0.0, 4.0, 0.0};
    double v1[3] = {2.0, 2.0, 1.0};
    double v2[3] = {1.0, 1.0, 1.0};

    double w0[3];
    double w1[3];
    double w2[3];
    vector_gso_normalization(v0,v1,v2,w0,w1,w2);

    print_vector(w0);
    print_vector(w1);
    print_vector(w2);
}
```