```
// ------------------------------------------------ fixed.c ------------------------------------------------
// Tianyun Duan
// Tahir Haideri

#include <stdint.h>
#include <stdio.h>
#include <math.h>
#include "ST7735.h"
#include "fixed.h"

int curXMin, curXMax, curYMin, curYMax, scaleFactorX, scaleFactorY;

#define scaleFactorX 128 / (curXMax-curXMin)
#define scaleFactorY 128 / (curYMax-curYMin)

void ST7735_sDecOut2(int32_t n){
	// check N's range
	if(n < -9999){
		printf("-**.**");
		return;
	}
	if(n > 9999){
		printf(" **.**");
		return;
	}

	// determine sign
	int positiveOrZero = (n >= 0);
	if(!positiveOrZero){
		n = -n;
		printf("-");
	}

	// solve for integral part
	int intPart = n / 100;
	printf("%d", intPart);
	printf(".");

	// solve for fractional part
	char hundredths = n % 10 + '0';
	char tenths = (n / 10) % 10 + '0';
	printf("%c", tenths);
	printf("%c", hundredths);
	return;
}

void ST7735_uBinOut6(uint32_t n){
	// check N's range
```

```
	if (n > 63999) {
		printf("***.**\n");
	}

	// solve for integral part
	int mask = 64;
	int integer = 0;
	int mtpl = 1;
	for (int i = 0; i < 10; i++) {
		integer += mtpl * ((mask & n) != 0);
		mtpl *= 2;
		mask <<= 1;
	}
	printf("%d.", integer);

	// solve for fractional part
	mask = 1;
	mtpl = 1;
	int frac = 0;
	for (int i = 0; i < 6; i++) {
		frac += mtpl * ((mask & n) != 0);
		mtpl *= 2;
		mask <<= 1;
	}
	if (frac != 0) {
		frac = frac * 100 / 64;
	}
	char tenths = (frac / 10) % 10 + '0';
	char hundredths = frac % 10 + '0';
	printf("%c", tenths);
	printf("%c", hundredths);
	return;
}

void ST7735_XYplotInit(char *title, int32_t minX, int32_t maxX, int32_t minY, int32_t maxY){
	ST7735_PlotClear(minY, maxY);
	ST7735_SetCursor(0,0);
	printf("%s", title);
	curXMin = minX;
	curXMax = maxX;
	curYMin = minY;
	curYMax = maxY;
}

void ST7735_XYplot(uint32_t num, int32_t bufX[], int32_t bufY[]){
	for(int i = 0; i < num; i++){
		int x = bufX[i];
		int y = bufY[i];
```

```
                if(x <= curXMax && x >= curXMin && y <= curYMax && y >= curYMin){
                        int realX = (x - curXMin) * 128 / (curXMax - curXMin);
                        int realY = 32 + (y - curYMin) * 128 / (curYMax - curYMin);
                        ST7735_DrawPixel(realX, realY, ST7735_BLUE);
                }
        }
}
```