小车控制程序清单

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 工程名称：SmartCar

\* 功能描述：结合飞思卡尔16位单片机MC9S12DG128B完成小车自动寻迹，沿黑线行驶功能

\* IDE环境： Metrowerks CodeWarrior 4.1

\* 组成文件：

\* main.c

\* SmartCar.c/PID.c/LCD1620.c/Test.c

\* 说明： 本版本为智能小车程序早期版本，还有待更进一步完善

\* 日期：2006-5-6

\* (c) Copyright 2006,Zhao Cheng

\* All Rights Reserved

\*

\*

\* By : Zhao Cheng

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* main.c

\*

\* (c) Copyright 2006,Zhao Cheng

\* All Rights Reserved

\*

\* By : Zhao Cheng

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <hidef.h> /\* common defines and macros \*/

#include <mc9s12dg128.h> /\* derivative information \*/

#pragma LINK\_INFO DERIVATIVE "mc9s12dg128b"

#define HIGHSPEED 11500 /\* 速度参量，此处未使用测速模块 \*/

#define LOWSPEED0 12500 /\* 0-24000 数值越大，速度越慢 \*/

#define LOWSPEED1 12000 /\* used in CarMain() \*/

#define STABMAX 50

#define StopCar() PORTK |= 0x80 /\* stop the motor \*/

#define StartCar() PORTK |= 0x04 /\* start the motor \*/

#define BrakeCar() PORTK &= 0xfb /\* slow the speed of the SmartCar \*/

unsigned int SYSCLOCK=0; /\* update in INT\_Timer0() \*/

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FUNCTION PROTOTYPES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* write in "SmartCar.c" \*/

void Init\_INT\_RTI(void); /\* initiate Real Time Interrupt \*/

void Init\_INT\_Timer(void); /\* INT\_Timer0 initiate \*/

void Init\_PWMout(void); /\* initiate PWM output \*/

void PWMout(int, int); /\* output PWM \*/

/\* write in "PID.c" \*/

void Init\_PID(void); /\* initiate PID parameter \*/

int CalculateP(void); /\* calculate parameter P \*/

float CalculatePID(void); /\* calculate PID \*/

int SignalProcess(unsigned char); /\* Process the signal from the sensors \*/

/\* write in "Test.c" \*/

void IOtest(void); /\* Test I/O \*/

void PWMtest(void); /\* Test PWM output \*/

int SignalTest(void); /\* Test the sensors \*/

/\* write in local file \*/

void Init(void); /\* initiate parameter \*/

void ProtectMoto(void); /\* the function protecting the Motor \*/

void CarMain(void); /\* SmartCar main function \*/

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 主程序

\*

\* 程序描述: 完成智能小车系统的初始化，通过按键可选择工作模式，有I/O测试，PWM 输出测试

\* 传感器测试，以及小车正常工作模式

\*

\* 硬件连接：PORTB 接传感器

\* PWM 输出口 （1）接舵机 （2）接电机驱动芯片MC33886

\*

\* 说明： 无

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void main(void)

{

Init();

DDRB = 0x00;

switch(PORTB)

{

case 0x80:

IOtest();

break;

case 0x40:

PWMtest();

break;

case 0x20:

SignalTest();

break;

default:

DDRA = 0x00;

DDRB = 0xff;

DDRK = 0xff;

PORTB = 0xff;

CarMain();

EnableInterrupts;

for(;;);

break;

}

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 小车寻迹行驶函数

\*

\* 程序描述: 通过传感器采集数据，并对其进行处理，通过PID算法得出小车稳定行驶所需的参数，进而调用PWM输出函数

\* 控制舵机与电机的工作

\*

\* 注意： 这个函数调用了 SignalProcess(unsigned char)，BrakeCar()，PWMout(Direction, Velocity)

\*

\* 说明： 无

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void CarMain(void)

{

static int Direction=0, Velocity;

static unsigned char signal;

static unsigned int BrakeTime = 0, BrakeControl = 0;

static unsigned int Stability=0, Stab[STABMAX], PStab=0, StabAver;

int i;

signal = PORTA;

PORTB = ~signal;

Direction = SignalProcess( signal );

/\* 稳定性系数的计算 \*/

Stability -= Stab[PStab];

Stab[PStab] = (unsigned int)Direction/100;

Stability += Stab[PStab];

PStab++;

if(PStab >= STABMAX) PStab=0;

StabAver = 0;

for(i=0;i<STABMAX;i++)

{

if(Stability > Stab[i])

StabAver += Stability - Stab[i];

else

StabAver += Stab[i] - Stability;

}

if( BrakeTime != 0)

{

BrakeTime--;

BrakeCar();

}

else

{

StartCar();

if(BrakeControl>0)

BrakeControl--;

if(Direction < -4000 || Direction > 4000 )

{

Velocity = LOWSPEED0;

if(BrakeControl == 0 && StabAver/STABMAX<22)

{

BrakeTime = 20;

BrakeControl = 120;

}

}

else

{

if(Direction < -2500 || Direction > 2500 )

Velocity = LOWSPEED1;

else

Velocity = HIGHSPEED;

}

}

PWMout(Direction, Velocity);

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 系统初始化函数

\*

\* 程序描述: 初始化了系统时钟，FLASH 和 EEPRO的工作频率，PWM输出口，定时器，以及PID算法中的有关参数

\*

\* 注意： 这个函数调用了 Init\_PWMout()nit\_INT\_Timer()nit\_PID()

\*

\* 说明： 无

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Init(void)

{

REFDV=0x01; /\* initiate PLL clock \*/

SYNR =0x02; /\* system clock 24M \*/

while (!(CRGFLG & 0x08)){} /\* wait untill steady \*/

CLKSEL=0x80; /\* 选定所相环时钟 \*/

FCLKDIV=0x49; /\* 使FLASH 和 EEPROM \*/

/\* 的擦除操作工作频率在200HZ左右 \*/

ECLKDIV=0x49;

Init\_PWMout(); /\* 01:50Hz 45:1kHz \*/

Init\_INT\_Timer(); /\* initiate ETC(Enhanced Capture Clock) \*/

Init\_PID(); /\* initiate PID caculating process \*/

DDRK |= 0x80; /\* Start Car -- stop car \*/

PORTK &= 0x7F;

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* SmartCar.c

\*

\* (c) Copyright 2006,Zhao Cheng

\* All Rights Reserved

\*

\* By : Zhao Cheng

\* Data : 2006\_5\_6

\* Note : Don't change this file if possible.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <hidef.h>

#include <mc9s12dg128.h>

extern SYSCLOCK; /\* 引用全局变量,系统时钟 \*/

void CarMain(void);

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* PWM初始化函数

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Init\_PWMout(void)

{

PWME = 0x22; /\*01:50Hz 45:1kHz \*/

PWMPOL = 0x22;

PWMCTL = 0x50;

PWMCLK = 0x02;

PWMSCLA = 4;

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* PWM输出函数

\* 程序描述：输入参数为方向，速度

\* 方向：－45～45

\* 速度：0～24000

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void PWMout(int Direction, int Velocity)

{

Direction = Direction/3 + 4500;

if(Direction<3000) Direction=3000;

if(Direction>6000) Direction=6000;

PWMPER01 = 60000; /\* Center 1500ms\*3 \*/

PWMDTY01 = Direction+93; /\* Set the steering angle \*/

if(Velocity>24000) Velocity=24000;

PWMPER45 = 24000; /\* 1kHz ( <10kHz ) \*/

PWMDTY45 = Velocity; /\* 设置电机速度 \*/

}

/\* initiate Real Time Interrupt 1.0 \*/

void Init\_INT\_RTI(void)

{

RTICTL = 0x74;

CRGINT |=0x80;

}

/\* Real Time Interrupt 1.0 \*/

interrupt void INT\_RTI(void)

{

CRGFLG |= 0x80; /\* clear the interrrupt flag \*/

}

/\* INT\_Timer0 initiate 1.0 \*/

void Init\_INT\_Timer(void)

{

TSCR2 =0x07; /\* 128Hz at 16M bus clok \*/

/\* 128Hz \* 2/3 at 24m bus clock \*/

/\* in fact it is a little more than it. \*/

TIOS |=0x01; /\* I/O select \*/

TIE |=0x01; /\* Interrupt Enable \*/

TSCR1|=0x80; /\* TSCR1\_TEN=1 //Timer Enable \*/

}

/\* INT\_Timer0 1.0 \*/

interrupt void INT\_Timer0(void)

{

SYSCLOCK++;

CarMain();

TC0 = TCNT + 1874; /\* 1875-1 :100Hz \*/

/\* F = Fosc / (TC\*128) \*/

TFLG1 |=0x01; /\* clear interrupt flag \*/

}

/\* not finished EEPROM \*/

void EEPROM(void)

{

ECLKDIV = 0x4F;

while(!(ECLKDIV&0x80)) /\* wheather \*/

{}

while(!(ESTAT&0x80)) /\* wheather the command buffer is empty \*/

{}

while(!(EPROT&0x80)) /\* wheather the eeprom is enabled to \*/

{}

ECMD = 0x41;

ESTAT |= 0x80;

while(!(ESTAT&0x80)) /\* wheather the command buffer is empty \*/

{}

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* PID.c

\* Description: This file includes some basic calculation function of PID

\* (c) Copyright 2006,Zhao Cheng

\* All Rights Reserved

\*

\* By : Zhao Cheng

\* Data : 2006\_5\_6

\* Note : Don't change this file if possible.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <mc9s12dg128.h> /\* derivative information \*/

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 宏定义

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define STABMAX 50

#define SENSORNUM 8

#define SAMPLETIMES 5

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FUNCTION PROTOTYPES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int CalculateP(void);

float CalculatePID(void);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* PID控制程序 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

struct CARSTATE

{

int E0;

int E1;

int E2;

int E3;

float Integral;

}CarState;

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 初始化PID参数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void Init\_PID()

{

CarState.E0 = 0;

CarState.E1 = 0;

CarState.E2 = 0;

CarState.E3 = 0;

CarState.Integral = 0;

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 信号处理函数

\*

\* 程序描述: 对传感器采集过来的数据进行处理，得到一些基本的计算参数

\*

\* 说明： 无

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int SignalProcess( unsigned int signal )

{

const int BitValue[8] = {43,26,12,6,-6,-12,-26,-43}; //MAX:28

int i,CurrPoint=0,LastPoint=0,BitNum=0;

unsigned char SignalBit[8];

for(i=0;i<8;i++)

{

SignalBit[i] = signal & 0x0001;

BitNum += SignalBit[i];

signal >>= 1;

}

switch(BitNum)

{

case 1:

for(i=0;i<8;i++)

if(SignalBit[i] != 0)

CurrPoint += BitValue[i];

CarState.E0 = CurrPoint;

break;

case 2:

for(i=0;i<8;i++)

if(SignalBit[i] != 0)

CurrPoint += BitValue[i];

CurrPoint >>= 1;

CarState.E0 = CurrPoint;

break;

default:

CarState.E0 = CarState.E1;

break;

}

return CalculateP()\*100;

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\* PID计算函数

\*

\* 程序描述: 计算P参数

\*

\* 说明： 无

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int CalculateP(void)

{

CarState.E1 = CarState.E0;

return((int)CarState.E0);

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* PID计算函数

\*

\* 程序描述: 对传感器采集过来的数据进行处理，得到一些基本的计算参数

\*

\* 说明： 无

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

float CalculatePID(void)

{

float P, I = 0, D;

/\* parameter const \*/

float Kp = 1.0, Ki = -0.0002, Kd = -0.0002;

/\* P parameter \*/

P = CarState.E0 \* Kp;

/\* I parameter \*/

if(P+I<2)

{

CarState.Integral += Ki \* CarState.E0;

I = CarState.Integral;

}

/\* D parameter \*/

D = Kd \* ( CarState.E0 + 3\*CarState.E1 - 3\*CarState.E2 - CarState.E3 )/6.0;

CarState.E3 = CarState.E2;

CarState.E2 = CarState.E1;

CarState.E1 = CarState.E0;

return (P+I+D);

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Test.c

\* Description: This file includes I/ O function for test, the PWM outputs function for test, function

\* testing sensors.

\* (c) Copyright 2006,Zhao Cheng

\* All Rights Reserved

\*

\* By : Zhao Cheng

\* Note : Don't change this file if possible.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <hidef.h>

#include <mc9s12dg128.h>

#define HIGHSPEED 8000

#define LOWSPEED 11000 /\* 速度变量，0-24000 数值越大，速度越慢 \*/

void PWMout(int, int);

/\* 24000-20000 \*/

void IOtest(void)

{

static unsigned char i=0,j=0x01,k;

DDRB = DDRA = 0xFF;

PORTB = 0xf0;

for(;;)

{

k=(~j)&0x7f;

PORTA = PORTB = k;

while (TCNT != 0x0000);

while (TCNT == 0x0000)

{

if(i>9)

{

j=j<<1;

i=0;

}

i++;

}

if(j>=0x80)

j=0x01;

}

}

void PWMtest(void)

{

int counter=-4500;

DDRB = 0xff;

PORTB = 0xff;

TSCR1 = 0x80; /\* enable timer TCNT \*/

TSCR2 = 0x00; /\* TCNT prescaler setup \*/

for(;;)

{

while (TCNT != 0x0000);

while (TCNT == 0x0000);

counter=counter+30;

if(counter >= 3000)

{

counter = 0;

PWMout(4500, LOWSPEED);

}

if(counter == 1500)

{

PWMout(-4500, LOWSPEED);

}

PORTB = (char)(counter/100);

}

}

void SignalTest(void)

{

unsigned char signal;

int Direction, Velocity;

Direction = 0;

Velocity = LOWSPEED;

DDRA = 0x00;

DDRB = 0xff;

signal = PORTA;

PORTB = ~signal;

switch(signal)

{

case 0x08: /\* 0001 1000 \*/

case 0x10:

Direction = 800;

Velocity = HIGHSPEED;

break;

case 0x04: /\* 0010 0100 \*/

case 0x20:

Direction = 1500;

Velocity = HIGHSPEED;

break;

case 0x02: /\* 0100 0010 \*/

case 0x40:

Direction = 2800;

Velocity = HIGHSPEED;

break;

case 0x01: /\* 1000 0001 \*/

case 0x80:

Direction = 4000;

Velocity = LOWSPEED;

break;

case 0x3c: /\* 0011 1100 over start line \*/

case 0xff: /\* 1111 1111 over crossing line \*/

case 0x00: /\* 0000 0000 go straight not need changed state \*/

default:

break;

}

if(signal > 0x0f)

Direction = -Direction;

PWMout(Direction, LOWSPEED);

}

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* LCD1620.c

\* ICC-AVR application builder : 2006-1-8 21:43:48

\* Target : M8

\* Crystal: 4.0000Mhz

\*

\* Note : Don't change this file if possible.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define CMD\_CLEAR 0x01

#define CMD\_RESET 0x02

#include <iom8v.h>

#include <macros.h>

#define LCD\_DATA 0xff

#define LCD\_EN 0x01 //PORTC 0

#define LCD\_RS 0x02 //PORTC 1

#define LCD\_RW 0x04 //PORTC 2

#define LCD\_DATAPORT PORTB

#define LCD\_ENPORT PORTA

#define LCD\_RSPORT PORTA

#define LCD\_RWPORT PORTA

void lcd\_init(void);

void lcd\_write\_cmd(unsigned cmd,unsigned data);

void lcd\_setxy(unsigned char x,unsigned char y);

void lcd\_write\_string(unsigned char X,unsigned char Y,unsigned char \*str);

void delay\_nus(unsigned int n);

void delay\_nms(unsigned int n);

void lcd\_init(void)

{

DDRB |= LCD\_DATA;

DDRA |= LCD\_EN | LCD\_RS | LCD\_RW;

LCD\_RWPORT&=~LCD\_RW;

LCD\_DATAPORT=0x30; //控制字规则：5：8bit,4:16x2,3:5x7

LCD\_ENPORT|=LCD\_EN;

delay\_nus(1);

LCD\_ENPORT&=~LCD\_EN;

delay\_nus(40);

lcd\_write\_cmd(0,0x38); //8bit test

lcd\_write\_cmd(0,0x0c); //显示开

lcd\_write\_cmd(0,0x01); //显示清屏

lcd\_write\_cmd(0,0x06); //显示光标移动设置

}

void lcd\_write\_cmd(unsigned cmd,unsigned data)

{

if(cmd==0)

LCD\_RSPORT&=~LCD\_RS;

else

LCD\_RSPORT|=LCD\_RS;

LCD\_DATAPORT&=0x00;

LCD\_DATAPORT=data;

LCD\_ENPORT|=LCD\_EN;

delay\_nus(10);

LCD\_ENPORT&=~LCD\_EN;

delay\_nus(10);

}

void lcd\_setxy(unsigned char x,unsigned char y)

{

unsigned char addr;

if(y==0)

addr=x+0x80;

else

addr=x+0xc0;

lcd\_write\_cmd(0,addr);

}

void lcd\_write\_string(unsigned char X,unsigned char Y,unsigned char \*str)

{

lcd\_setxy(X,Y);

while(\*str)

{

lcd\_write\_cmd(1,\*str);

str++;

}

}

void delay\_1us(void) //1us延时函数

{

asm("nop");

}

void delay\_nus(unsigned int n) //N us延时函数

{

unsigned int i=0;

for (i=0;i<n;i++)

delay\_1us();

}

void delay\_1ms(void) //1ms延时函数

{

unsigned int i;

for (i=0;i<1140;i++);

}

void delay\_nms(unsigned int n) //N ms延时函数

{

unsigned int i=0;

for (i=0;i<n;i++)

delay\_1ms();

}

//call this routine to initialize all peripherals

void main(void)

{

lcd\_init();

while(1)

{

lcd\_write\_cmd(0,0x01); //清屏

delay\_nms(2);

lcd\_write\_string(0,0,"happy new year");

delay\_nms(100);

lcd\_write\_string(0,1,"LCD successful!");

delay\_nms(100);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 程序结束 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/