Autonomous/Remote
Controlled Robot Project Report

Carlos Jaime Collazo
EEL5666
Dr. Antonio Arroyo
Intelligent Machines Design Laboratory
University of Florida
August 5, 1998

Table of Contents

Abstract

Following is a discussion on developing an obstacle avoidance system, sonar guided system, and remote control and autonomous control mode switching capabilities for a high speed hobby race car (RC) using infrared emitter and sensors, sonar transducers and a 75MHz AM transmitter/receiver pair. The RC called Tuco uses a commercial speed control servo to control the high current motor. RF noise produced by the motors and the micro-controller complicated the autonomous mode of operation.

Executive Summary

Tuco is an autonomous slave RC that has the ability to travel at high speeds. A commercial RC platform was used to mount a Motorola 68HC11 microprocessor (MP), the motor, speed control device (SC), steering servo (SS) and differential gear. Tuco has actuators such as RC size shock absorbers to cushion the frame when in motion. Although Tuco is capable of running at about 50mph, the speed controller was used in conjunction with the MP board to maintain low speeds allowing Tuco perform obstacle avoidance operations with IR emitters and receivers. The speed controller allows for forward and reverse motor direction control modes. Two IR receivers are placed at the front for forward motion and two in the back for obstacle avoidance when reversing.

At lower speeds, Tuco behaves more efficiently when utilizing the sonar system capabilities. Three sonar receivers are installed in the platform for Tuco to follow a moving beacon: a hand held 40kHz sonar transmitter. The receivers are placed at the front of the RC in a triangular form. The robot translates the loudness (signal amplitude) of the three received signals and follows the direction of the loudest of the three. The idea behind the inclusion of a sonar tracking device is to have Tuco follow the transmission in the way a canine would follow his/her master when called.

The MP utilizes its input capture features to process the two control pulses received by the 75MHz AM receiver installed in the robot. One of the signals controls the steering servo and the other controls the speed and motion direction. Upon reception of the pulse width modulated control signals, the MP decides whether or not the remote control signal has changed in pulse width before changing the mode of operation (remote control and autonomous).

Introduction

The primary concern with Tuco is to have it find a sonar transmission beacon like a canine would follow the master during autonomous mode. In order to accomplish such a task, three sonar transducers connected to a 40kHz band pass filter (BPF) are used. Another concern is to have Tuco respond to the RF transmission as it would normally do without any micro-controllers in the system.

Integrated System

Tuco is a four wheel RC utilizing the MP 32Kbyte memory expansion and motor port board designed by Scott Jantz. To avoid the noise coming from the motors and the MP a Faraday Shield composed of a copper sheet connected to the common ground of the system was used. As the SC and the SS are connected to the Output Compare, and the SC and SS signals coming from the RF receiver are connected directly into the Input Capture ports of the MP, if a change in the signal pulse widths are detected beyond (10 clock cycle counts, the MP overrides the autonomous operation and gives to remote control mode.

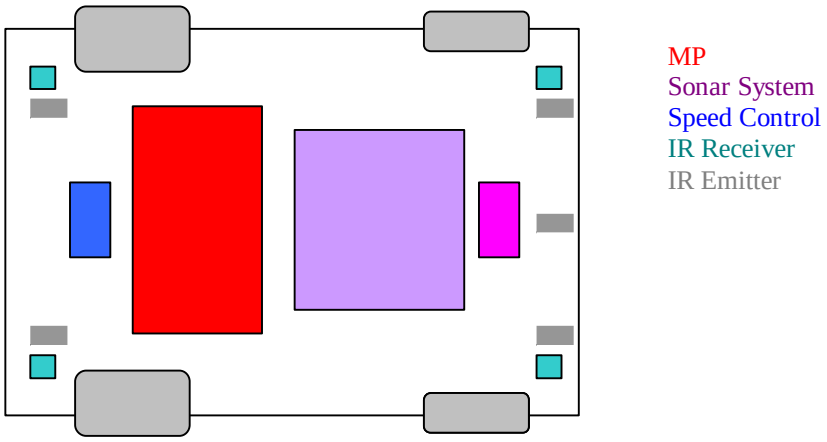
The four IR receiver and the three sonar transducer processed analog signals are converted to digital by seven of the Analog to Digital (A/D) ports of the MP. When the digital signal translated as a numerical value is greater than a threshold, the MP is programmed to respond and behave in certain ways. For instance, when the two from IR receiver signals pass the threshold, the MP stops, steers against the obstacle and reverses for about 2 seconds. After that,Tuco searches for a change in threshold in the two rear IR receivers for obstacle avoidance. Once this happens,

Tuco resumes its normal mode of autonomous sonar guided operation. When the sonar signals surpass the threshold, the MP determines the maximum signal out of the three transducers and follows the direction of the source. The transducers are placed in the following order: left, center and right.

Mobile Platform

Tuco is a four wheel RC and has the layout shown in Figure 1.

Figure 1



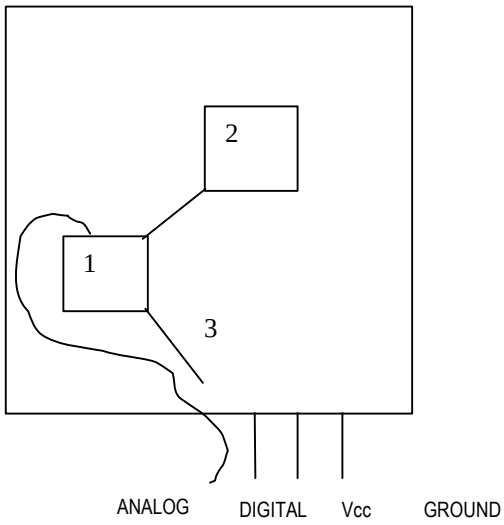
Actuation

Tuco uses two commercial actuators: Futuba steering servo and speed control. Both response to pulse modulated signals driven by TTL level (+5V) amplitudes. The period for each is around 20ms and the pulse widths range from 0.7ms for left or reverse and 2ms for right or forward. The MP controls these devices directly from Output Compare ports two and three.

Sensors

The IR receiver analog outputs are connected directly into the analog ports of the MP as mentioned above. The receivers are the hacked version of the Sharp GP1U5 IR receivers. Figure 2 shows the design and hacking method of the Sharp IR receivers.

Figure 2



Basically, trace 3 above was cut and a wire was soldered to the top of box 1 which is a 0.1uF capacitor to have access to analog readings from the IR receivers. Box 2 is the integrated section.

The sonar system design is shown in the Appendix.

Behaviors

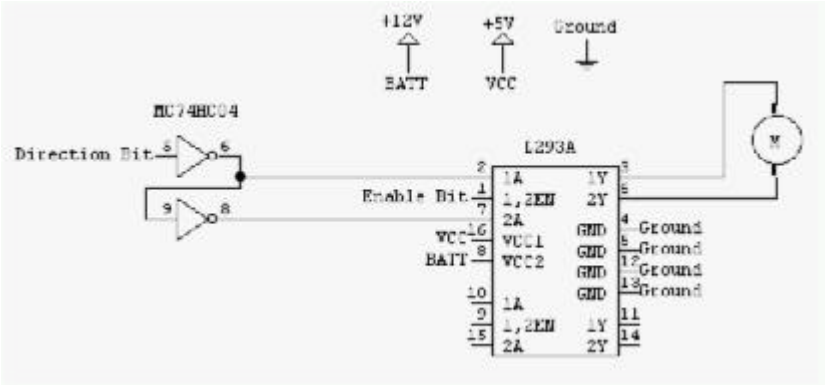
As mentioned above, in autonomous mode Tuco finds the maximum value out of the three signals coming from the sonar transducers and follows the direction of the transducer in the corresponding position. Simultaneously, Tuco avoids obstacles when following the beacon. Observe the subroutines AVOID and SONAR in theAppendix.

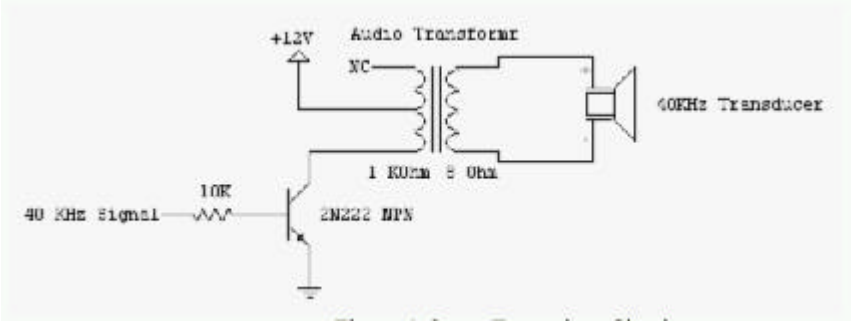
Conclusion

Tuco follows the moving beacon, avoids obstacles and switches between operation modes as expected. All systems performed properly.

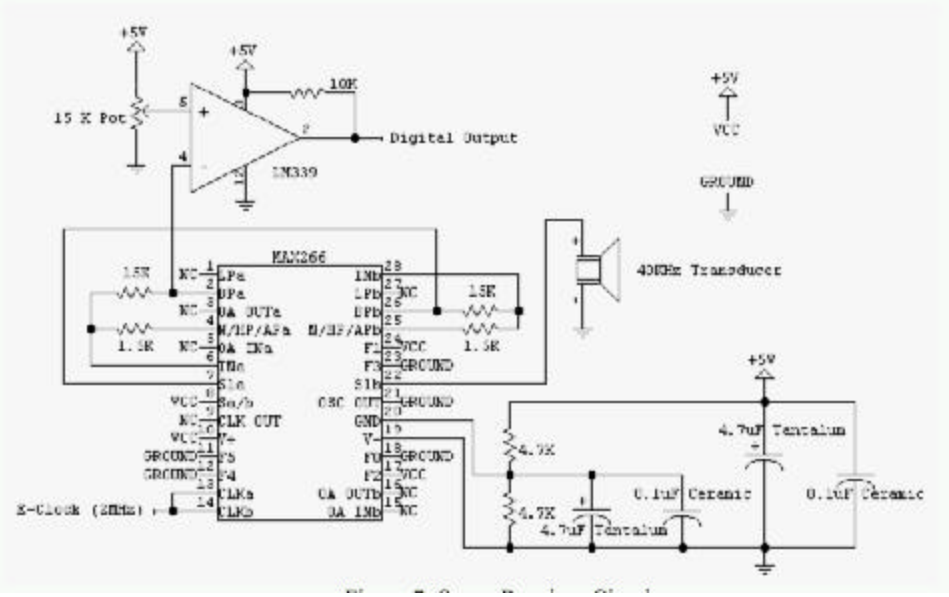
Appendix

Sonar Transmitter Circuitry





Sonar Receiver Circuitry



Behavior Code

****

```
* EEL5666
* Tuco's Brain
* Carlos Jaime Collazo
*
*       OC2=Steering Servo=PIN28
*       OC3=Speed Control=PIN29
*       A1=Used by Buffalo!!
*       A2=GREEN=RIGHT_IR=PIN45=PE1
*       A3=BLUE=LEFT_IR=PIN47=PE2
*       A4=Sonar Output2=Center=PE3=PIN49
*       A5=Sonar Output3=Left=PE4=PIN44
*       A6=Right Rear IR=PE5=PIN46
*       A7=Left Rear IR=PE6=PIN48
*       A8=Sonar Output1=Right=PE7=PIN50
*       IR => Vref High = 5V, Vref Low = 0V
*       VRHIGH=PIN52;VRLOW=PIN51
*       IC1=PIN32=CHANNEL1=STEERING, IC2=PIN33=CHANNEL2=SPEED C.
*
* SAMPLE IC OUTPUTS
*
* IC1=CHANNEL1
* PW1 :  L1:  P1:CTR1:MIN1:MAX1
* 0BA7:8A5D:9606:0BAB:087C:0EE2
* IC2=CHANNEL2
* PW2 :  L2:  P2:CTR2:MIN2:MAX2
* 0BEC:8A17:9605:0BF1:0852:0F69
* CENTER VALUES BEFORE ANY STEERING OR SPEED CHANGE:
* 0C14:8A03:9615:0C0A:0C08:0C20
* 0C0C:8A13:9618:0C21:0C00:0C13
* range: ee7-c12=2d5=725base10
************************************************************


*---------------------------------------
*       EQUATES/REGISTERS
*---------------------------------------

*_____MEMORY_____________________________
TSTACK  EQU     $01FF   ;Top of Stack
STACK   EQU     $0041   ;STACK
BASE    EQU     $1000   ;BEGINNING OF REGISTERS
RAM     EQU     $0100   ;RAM address
SRAM            EQU             $8000           ;Low Addr. Static RAM
EEPROM  EQU     $B600   ;EEPROM
START           EQU             $F800           ;PCBUG11 START
*_____TIMER_____________________________
TMSK1   EQU     $22     ; TIMER MASK1 REGISTER
TFLG1   EQU     $23     ; TIMER FLAG1 REGISTER
TMSK2   EQU     $24     ; TIMER MASK2 REGISTER
TFLG2   EQU     $25     ; TIMER FLAG2 REGISTER
TCTL1   EQU     $20     ; TIMER CONTROL 1
TCTL2   EQU     $21     ; TIMER CONTROL 2
TCNT    EQU     $0E     ; TCNT High byte
*_____I/O PORTS_________________________
PORTA   EQU     $00
PORTB   EQU     $04
PORTC   EQU     $03
PORTCL  EQU     $05
PORTD   EQU     $08
PORTE   EQU     $0A
DDRC    EQU     $07
DDRD    EQU     $09
ME11IO          EQU     $7000
*_____INPUT CAPTURE_____________________
TIC1    EQU     $10     ; Timer Input Capture 1
TIC2    EQU     $12     ; Timer Input Capture 2
TIC3    EQU     $14     ; Timer Input Capture 3
*_____OUTPUT CAPTURE____________________
TOC1    EQU     $16     ;OUTPUT COMPARE 1 REGISTER
TOC2    EQU     $18     ;OUTPUT COMPARE 2 REGISTER
```

```
TOC3    EQU     $1A     ;OUTPUT COMPARE 3 REGISTER
TOC4    EQU     $1C     ;OUTPUT COMPARE 4 REGISTER
*_____A/D SYSTEM_____________________
*VRHIGH=PIN52;VRLOW=PIN51
OPTION  EQU     $39     ;SYSTEM CONFIG. OPTIONS
ADCTL   EQU     $30     ;A/D CONTROL/STATUS
ADR1    EQU     $31     ;ANALOG PORT1:
ADR2    EQU     $32     ;ANALOG PORT2:
ADR3    EQU     $33     ;ANALOG PORT3:
ADR4    EQU     $34     ;ANALOG PORT4:
ADR5    EQU     $31     ;ANALOG PORT5:
ADR6    EQU     $32     ;ANALOG PORT6:
ADR7    EQU     $33     ;ANALOG PORT7:
ADR8    EQU     $34     ;ANALOG PORT8:
* AN0=PE0=PIN43; AN1=PE1=PIN45
* AN2=PE2=PIN47; AN3=PE3=PIN49
* AN4=PE4=PIN44; AN5=PE5=PIN46
* AN6=PE6=PIN48; AN7=PE7=PIN50
*_____PULSE ACCUMULATOR_____________________
PACTL   EQU     $26     ; PA Control
PACNT   EQU     $27     ; PA Counter
*_____SCI___________________________________
BAUD    EQU     $102B   ; BAUD rate control register to set the BAUD rate
SCCR1   EQU     $102C   ; Serial Communication Control Register-1
SCCR2   EQU     $102D   ; Serial Communication Control Register-2
SCSR    EQU     $102E   ; Serial Communication Status Register
SCDR    EQU     $102F   ; Serial Communication Data Register
*_____Interrupt Vectors_____________________
BIC1I   EQU     $E8     ;BUFFALO INT. VECTOR IC1
BIC2I   EQU     $E5     ;BUFFALO INT. VECTOR IC2
BOC2I   EQU     $DC     ;BUFFALO INT. VECTOR OC2
BOC3I   EQU     $D9     ;BUFFALO INT. VECTOR OC3
BATODI  EQU     $EB     ;BUFFALO INT. VECTOR RTI FOR A/D
OC2I    EQU     $FFE6   ;INT. VECTOR OC2
OC3I    EQU     $FFE4   ;INT. VECTOR OC3
ATODI   EQU     $FFF0   ;RTI FOR A/D
RESET   EQU     $FFFE   ;Reset interrupt vector


*---------------------------------------
*       CHARACTERS/CONSTANTS
*---------------------------------------

EOS     EQU     $04     ; User-defined End Of
*                       ; String (EOS) character
CR      EQU     $0D     ; Carriage Return Character
LF      EQU     $0A     ; Line Feed Character
ESC     EQU     $1B     ; Escape Charracter
QTE     EQU     $27     ; QUOTE '
SCLN    EQU     $3B     ; SEMICOLON ;
ZERO    FCB     '0'
        FCB     '1'
        FCB     '2'
        FCB     '3'
        FCB     '4'
        FCB     '5'
        FCB     '6'
        FCB     '7'
        FCB     '8'
        FCB     '9'
        FCB     'A'
        FCB     'B'
        FCB     'C'
        FCB     'D'
        FCB     'E'
        FCB     'F'
ANA0    FCC     @Left:@
        FCB     EOS
ANA1    FCC     @Right@
        FCB     EOS
COL     FCC     @:@
```

```
        FCB     EOS
CRLF    FCB     CR, LF, EOS
CLR     FCB     ESC,$5B,$32,$4A       ; ANSI sequence to clear screen
        FCB     EOS             ; EOS character
POS     FCB     ESC,$5B,$3B,$48       ;     and move cursor to home
        FCB     EOS             ; EOS character
PW1PRNT FCC     @PW1=@
        FCB     EOS             ; EOS character
L1PRNT  FCC     @L1=@
        FCB     EOS             ; EOS character
P1PRNT  FCC     @P1=@
        FCB     EOS             ; EOS character
PW2PRNT FCC     @PW2=@
        FCB     EOS             ; EOS character
L2PRNT  FCC     @L2=@
        FCB     EOS             ; EOS character
P2PRNT  FCC     @P2=@
        FCB     EOS             ; EOS character


*---------------------------------------
*       BIT MASKS
*---------------------------------------

BIT0    EQU     %00000001
BIT1    EQU     %00000010
BIT2    EQU     %00000100
BIT3    EQU     %00001000
BIT4    EQU     %00010000
BIT5    EQU     %00100000       ;OC3
BIT6    EQU     %01000000
BIT7    EQU     %10000000       ;OC2
INV0    EQU     %11111110
INV1    EQU     %11111101
INV2    EQU     %11111011
INV3    EQU     %11110111
INV4    EQU     %11101111
INV5    EQU     %11011111       ;OC3
INV6    EQU     %10111111
INV7    EQU     %01111111       ;OC2
BIT74   EQU     %10010000
BIT65   EQU     %01100000
BIT54   EQU     %00110000


*---------------------------------------
*       Data
*---------------------------------------

MID     EQU     $71     ;Middle: ~(2^8)/2
ADF     EQU     0
IOPAT   EQU     $FF     ;All PORTD outputs
DUMMYP  EQU     $0C20   ;DUMMY CENTER
DUMMYL  EQU     $9606-$0C20     ;DUMMY CENTER DIFF
DUMMYMIN        EQU     $900    ;DUMMY MINIMUM
DUMMYMAX        EQU     $F90    ;DUMMY MAXIMUM
*TOLLERANCE    EQU     $A      ;TOLLERANCE VALUE
TOLLERANCE      EQU     $14     ;TOLLERANCE VALUE
THRESHOLD       EQU     $18     ;SERVO CONTROL THRESHOLD
*OFFSET EQU     $38
OFFSET  EQU     $D

SLOWF   EQU     $DF+OFFSET      ;ADD OR SUB FOR SLOW MOTION (FWD)
*decrease pulsewidth to go fwd, incrase to go backwards.
SLOWR   EQU     $57+OFFSET      ;ADD OR SUB FOR SLOW MOTION (REV)
LEVEL   EQU     $A8     ;STEER LEVEL UNIT (5 LEVELS FROM CENTER
*                       TO AN EXTREME).
SONART  EQU     $F
```

```
*---------------------------------------
*       INTERRUPT VECTOR ADDRESSES
*---------------------------------------

        ORG     BOC2I
*       FDB     OC2IS
        JMP     OC2IS

        ORG     BOC3I
*       FDB     OC3IS
        JMP     OC3IS


        ORG     BIC1I
        JMP     IC1ISR

        ORG     BIC2I
        JMP     IC2ISR


*       ORG     BATODI
*       JMP     ADSERV

*       ORG             RESET
*       FDB             MAIN


*---------------------------------------
*       Variables
*---------------------------------------

        ORG     RAM
        JMP     MAIN

        ORG     SRAM
A1      RMB     1       ; BUFFALO ANALOG PORT
RFA2    RMB     1       ; RF ANALOG PORT
LFA3    RMB     1       ; LF ANALOG PORT
SI2A4   RMB     1       ; SI2 ANALOG PORT
SI3A5   RMB     1       ; SI3 ANALOG PORT
RRA6    RMB     1       ; RR ANALOG PORT
LRA7    RMB     1       ; LR ANALOG PORT
SI1A8   RMB     1       ; SI1 ANALOG PORT
PAN2    RMB     1       ; PREVIOUS ANALOG PORT TMP
PAN3    RMB     1       ; PREVIOUS ANALOG PORT TMP
ADCNT   RMB     1       ; A/D System Counter
PW1     RMB     2       ; Pulse Width Register 1
PW2     RMB     2       ; Pulse Width Register 2
L1      RMB     2       ; Low part of pulse 1
L2      RMB     2       ; Low part of pulse 2
P1      RMB     2       ; PERIOD 1
P2      RMB     2       ; PERIOD 2
PPW1    RMB     2       ; PREVIOUS PULSE WIDTH 1
PPW2    RMB     2       ; PREVIOUS PULSE WIDTH 2
PWA1    RMB     2       ; Pulse Width Register 1
PWA2    RMB     2       ; Pulse Width Register 2
LA1     RMB     2       ; Low part of pulse 1
LA2     RMB     2       ; Low part of pulse 2
PA1     RMB     2       ; PERIOD 1
PA2     RMB     2       ; PERIOD 2
T       RMB     2       ; Delay max
RTMP    RMB     2       ; Temporary register
LTMP    RMB     2       ; Temporary register
FTMP    RMB     2       ; Forward temp
RVTMP   RMB     2       ; Reverse temp
ACCELF  RMB     2       ; ACCELERATION TEMP
DECELF  RMB     2       ; DECELERATION LOOP FACTOR
STEER   RMB     2       ; STEERING TEMPORARY
ACCEL   RMB     2       ; ACCELERATION TEMP
TURN    RMB     2       ; TURN TIME RANGE FACTOR
DBUFR   RMB     5       ; 5 BYTE VARIABLE FOR HTOD SRTINE.
```

```
WORD    RMB     2       ; TEST WORD
BYTE    RMB     1       ; TEST BYTE
*INPUT CAPTURE VALUES
MAX1    RMB     2       ; MAX PULSE WIDTH CHANNEL1 (FULL RIGHT)
MAX2    RMB     2       ; MAX PULSE WIDTH CHANNEL2 (FULL REV)
MIN1    RMB     2       ; MIN PULSE WIDTH CHANNEL1 (FULL LEFT)
MIN2    RMB     2       ; MIN PULSE WIDTH CHANNEL2 (FULL FWD)
CENTER1 RMB     2       ; CENTER CHANNEL1
CENTER2 RMB     2       ; CENTER CHANNEL2
SLOWFWD RMB     2       ; SLOW FORWARD VALUE
SLOWREV RMB     2       ; SLOW REVERSE VALUE
TOHI1   RMB     2       ; IC1 LO TO HI TCNT VALUE
TOHI2   RMB     2       ; IC2 LO TO HI TCNT VALUE
TOLO1   RMB     2       ; IC1 HI TO LO TCNT VALUE
TOLO2   RMB     2       ; IC2 HI TO LO TCNT VALUE
PTOHI1  RMB     2       ; PREVIOUS IC1 LO TO HI TCNT VALUE
PTOHI2  RMB     2       ; PREVIOUS IC2 LO TO HI TCNT VALUE
REMOTE  RMB     1       ; REMOTE ON/OFF SWITCH
REMOTE1 RMB     1       ; REMOTE ON/OFF SWITCH
REMOTE2 RMB     1       ; REMOTE ON/OFF SWITCH
TEMP1   RMB     2
TEMP2   RMB     2
OC2LTH  RMB     2       ; OC2 LO TO HI TCNT VALUE
OC3LTH  RMB     2       ; OC3 LO TO HI TCNT VALUE
LTHDIFF2        RMB     2       ; OC2 LO TO HI DIFF. = PERIOD
LTHDIFF3        RMB     2       ; OC3 LO TO HI DIFF. = PERIOD
OC2HTL  RMB     2       ; OC2 HI TO LO TCNT VALUE
OC3HTL  RMB     2       ; OC3 HI TO LO TCNT VALUE
MIDL    RMB     1       ; LEFT ANALOG MIDDLE VALUE (A2)
MIDR    RMB     1       ; RIGHT ANALOG MIDDLE VALUE (A3)
AVDL    RMB     1       ; AVOID LEFT SWITCH
AVDR    RMB     1       ; AVOID RIGHT SWITCH
TOLLERANCE1     RMB     2       ; PULSE 1 TOLLERANCE
TOLLERANCE2     RMB     2       ; PULSE 2 TOLLERANCE
SI1MIN  RMB     1       ; SONAR INPUT 1 MIN
SI2MIN  RMB     1       ; SONAR INPUT 2 MIN
SI3MIN  RMB     1       ; SONAR INPUT 3 MIN
S1      RMB     1       ; SONAR TEMP 1
S2      RMB     1       ; SONAR TEMP 2
S3      RMB     1       ; SONAR TEMP 3
PS1     RMB     1       ; SONAR PREVIOUS TEMP 1
PS2     RMB     1       ; SONAR PREVIOUS TEMP 2
PS3     RMB     1       ; SONAR PREVIOUS TEMP 3
TS1     RMB     1       ; SONAR PREVIOUS TEMP 1
TS2     RMB     1       ; SONAR PREVIOUS TEMP 2
TS3     RMB     1       ; SONAR PREVIOUS TEMP 3
SCTRL   RMB     1       ; SONAR CONTROL VARIABLE
SMAX    RMB     1       ; SONAR MAX


*********************************************************
*
*       MAIN
*
*********************************************************


MAIN            LDS     #TSTACK ;Init. stack at $1ff ($ff total)
*                       ;  need more than $41 for printing
        LDX     #BASE   ;Init. register base
*       JSR     InitSCI ;Init. SCI Port
        JSR     TIMER   ;Init. Timer
        JSR     IC12INIT        ;Init. IC1 and IC2
        JSR     DELAY
        JSR     OC2INIT ;" OC2
        JSR     OC3INIT ;" OC3
        JSR     ADINIT  ;Init. A/D system
        JSR     INIT_COUNT      ;INITIALIZE COUNTER
        JSR     CLRS    ;Clear screen
        JSR     DUMMYC  ;INIT DUMMY CONTROLS
```

```
        CLI             ;Enable Interrupts

        JSR     LDELAY  ;ALLOW DATA STORING
        JSR     ADNOI   ;GET A/D VALUES
        JSR     DELAY   ;ALLOW DATA STORING
        JSR     MINMAX1 ;CALCULATE INITIAL CHNL. VALUES
        JSR     MINMAX2
        JSR     INIT_SERVOS ;INITIALIZE SERVOS
        JSR     INIT_IR ;INITIALIZE IR SYSTEM
        JSR     INIT_SONAR      ;INITIALIZE SONAR REGISTERS


*---------------------------------------
*       BODY
*---------------------------------------

        CLRA    REMOTE  ;INITIALIZE CONTROL TO AUTONOMOUS

PARALLEL        NOP             ;CHECK IF BOTH ARE RECEIVING

        JSR     SMAXCAL
        JSR     SONARCHK
*       JSR     SPRINT

        BRA     PARALLEL


*---------------------------------------
*       CONTROL SUBROUTINES
*---------------------------------------

*-----------------------
* SONAR SYSTEM
*-----------------------


*-----------------------
SONARCHK        NOP
        LDAA    SCTRL
        CMPA    #1
        BEQ     TRNR
        LDAA    SCTRL
        CMPA    #3
        BEQ     TRNL
        BRA     SCNTR
TRNR    NOP
        JSR     FCTOR   ;STEER RIGHT
        JSR     SFWD
        JSR     DELAY
        JSR     DELAY
        JSR     AVOID
        JSR     DELAY
*       JSR     SPRINT
        JSR     RTOC
        JSR     DELAY
        JSR     SREV
        JSR     STOPREV
        BRA     SCHKDONE
TRNL    NOP
        JSR     FCTOL   ;STEER RIGHT
        JSR     SFWD
        JSR     DELAY
        JSR     DELAY
        JSR     AVOID
*       JSR     DELAY
*       JSR     SPRINT
        JSR     LTOC
        JSR     DELAY
        JSR     SREV
```

```
        JSR     STOPREV
        BRA     SCHKDONE
SCNTR   NOP
        JSR     SFWD
        JSR     AVOID
*       JSR     DELAY
*       JSR     SPRINT
        JSR     SREV
        JSR     STOPREV
SCHKDONE        RTS

*-----------------------
SPRINT  NOP
        LDAA    #'S'
        JSR     ochar
        LDAA    #'1'
        JSR     ochar
        CLRA
        LDAB    S1
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'2'
        JSR     ochar
        CLRA
        LDAB    S2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'3'
        JSR     ochar
        CLRA
        LDAB    S3
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'C'
        JSR     ochar
        CLRA
        LDAB    SCTRL
        JSR     HXTOA

****************************************
        LDX     #CRLF
        JSR     PrintS
        LDX     #POS
        JSR     PrintS
        RTS
*-----------------------
* SONAR MIN. VALUES
*-----------------------

SMINS   LDAA    SI1A8
        CMPA    SI1MIN
        BHS     CHKSI2
        STAA    SI1MIN
CHKSI2  LDAA    SI2A4
        BHS     CHKSI3
        STAA    SI2MIN
CHKSI3  LDAA    SI3A5
        BHS     SMINDONE
        STAA    SI3MIN
```

```
SMINDONE       RTS


*-----------------------
* SONAR DIFFERENCES
*-----------------------

SDIFF   NOP
        LDAA    SI1A8
        STAA    TS1
        SUBA    PS1
        STAA    S1
        BLE     NEXTS1
        BRA     SDONE
NEXTS1  LDAA    SI2A4
        STAA    TS2
        SUBA    PS2
        STAA    S2
        BLE     NEXTS2
        BRA     SDONE
NEXTS2  LDAA    SI3A5
        STAA    TS3
        SUBA    PS3
        STAA    S3
SDONE   LDAA    TS1
        STAA    PS1
        LDAA    TS2
        STAA    PS2
        LDAA    TS3
        STAA    PS3
        RTS


*-----------------------
* SONAR MAX CALC
*-----------------------

SMAXCAL LDAA    #0
        STAA    SMAX
        LDAA    S1
        CMPA    SMAX
        BLT     NM2
        STAA    SMAX
        LDAA    #1
        STAA    SCTRL
NM2     LDAA    S2
        CMPA    SMAX
        BLT     NM3
        STAA    SMAX
        LDAA    #2
        STAA    SCTRL
NM3     LDAA    S3
        CMPA    SMAX
        BLT     NMDONE
        STAA    SMAX
        LDAA    #3
        STAA    SCTRL
NMDONE  NOP
        RTS


*-----------------------
* OBSTACLE AVOIDANCE
*-----------------------

AVOID   LDAA    AVDL
        ADDA    AVDR
        BEQ     AVDDONE
        JSR     STOPFWD ;Stop motor
        LDAA    AVDL
        BEQ     CHKR
```

```
        JSR     AVOIDL
        BRA     AVDDONE
CHKR    LDAA    AVDR
        BEQ     AVDDONE
        JSR     AVOIDR
AVDDONE RTS


*-----------------------
* OBSTACLE DETECTION
* CHECK
*-----------------------

CHECK   LDAA    LFA3    ;CHECK LEFT
        CMPA    MIDL    ;COMPARE WITH MIDDLE
        BLO     NOLEFT
AVOID_LEFT      LDAA    #1
        STAA    AVDL
        BRA     CHK_R
NOLEFT  CLRA
        STAA    AVDL
CHK_R   LDAA    RFA2
        CMPA    MIDR    ;COMPARE WITH MIDDLE
        BLO     NORIGHT
AVOID_RIGHT     LDAA    #1
        STAA    AVDR
        BRA     CHKDONE
NORIGHT CLRA
        STAA    AVDR
CHKDONE RTS


*-----------------------
* OBSTACLE AVOIDANCE
* INDEPENDENT CONTROLS
*-----------------------

AVOIDR  NOP
        JSR     FCTOR   ;STEER RIGHT
        JSR     SREV
        JSR     LDELAY  ;REVESE DELAY
        JSR     STOPREV
        JSR     RTOC
*       JSR     SFWD
        RTS

AVOIDL  NOP
        JSR     FCTOL   ;STEER LEFT
        JSR     SREV
        JSR     LDELAY
        JSR     STOPREV
        JSR     LTOC
*       JSR     SFWD
        RTS


*-----------------------
* MOTOR CONTROL
*-----------------------


SFWD    LDD     SLOWFWD
        LDX     ACCELF
        JSR     FWD
        JSR     DELAY
        JSR     DELAY
        RTS

STOPFWD LDD     CENTER2
        LDX     DECELF
        JSR     REV
```

```
        RTS

SREV    LDD     SLOWREV
        LDX     ACCELF
        JSR     DELAY
        JSR     DELAY
        JSR     REV
        RTS

STOPREV LDD     CENTER2
        LDX     DECELF
        JSR     FWD
        RTS

FWD     STD     FTMP
FLOOP   NOP
        JSR     XDELAY
        LDD     PWA2
        SUBD    ACCEL
        STD     PWA2
        JSR     L2CAL
        CPD     FTMP
        BNE     FLOOP
        RTS

REV     STD     RTMP
RLOOP   NOP
        JSR     XDELAY
        LDD     PWA2
        ADDD    ACCEL
        STD     PWA2
        JSR     L2CAL
        CPD     RTMP
        BNE     RLOOP
        RTS

L2CAL   PSHA
        PSHB
        LDD     P2
        SUBD    PWA2
        STD     LA2
        PULB
        PULA
        RTS



*-----------------------
* STEER CONTROL
*-----------------------

CTOL    ADDD    CENTER1 ;CENTER TO LEFT SPECIFIED IN D
        JSR     LEFT
        RTS

CTOR    ADDD    CENTER1 ;CENTER TO RIGHT SPECIFIED IN D
        JSR     RIGHT
        RTS

RTOC    LDD     CENTER1 ;RIGHT TO
        JSR     LEFT    ;CENTER
        RTS

FCTOL   LDD     MIN1    ;CENTER TO
        JSR     LEFT    ;FULL LEFT
        RTS

LTOC    LDD     CENTER1 ;LEFT
        JSR     RIGHT   ;TO CENTER
        RTS
```

```
FCTOR   LDD     MAX1    ;CENTER TO
        JSR     RIGHT   ;FULL RIGHT
        RTS

LEFT    STD     LTMP
LLOOP   LDD     PWA1
        SUBD    STEER
        STD     PWA1
        JSR     L1CAL
        CPD     LTMP
        BNE     LLOOP
        RTS

RIGHT   STD     RTMP
RLOOP   LDD     PWA1
        ADDD    STEER
        STD     PWA1
        JSR     L1CAL
        CPD     RTMP
        BNE     RLOOP
        RTS

L1CAL   PSHA
        PSHB
        LDD     P1
        SUBD    PWA1
        STD     LA1
        PULB
        PULA
        RTS


*-----------------------
* SERVO MIN AND MAX
* VALUES CALCULATIONS
*-----------------------

MINMAX1 LDD     PW1
        CPD     MIN1
        BHI     CMM11
        STD     MIN1
CMM11   CPD     MAX1
        BLO     CMM12
        STD     MAX1
CMM12   RTS


MINMAX2 LDD     PW2
        CPD     MIN2
        BHI     CMM21
        STD     MIN2
CMM21   CPD     MAX2
        BLO     CMM22
        STD     MAX2
CMM22   RTS


*---------------------------------------
*       PRINT IC AND OC VALUES
*---------------------------------------

SEGMENT1        NOP
        LDD     PW1
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     L1
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     P1
```

```
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        RTS

SEGMENT2        NOP
        LDD     CENTER1
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     MIN1
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     MAX1
        JSR     HXTOA
        LDX     #CRLF
        JSR     PrintS
        RTS

SEGMENT3        NOP
        LDD     PW2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     L2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     P2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        RTS

SEGMENT4        NOP
        LDD     CENTER2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     MIN2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     MAX2
        JSR     HXTOA
        LDX     #CRLF
        JSR     PrintS
        RTS

*---------------------------------------
*       Print analog port values
*---------------------------------------

SEGMENT5        NOP
        LDX     #BASE

****************************************
        LDAA    #'R'
        JSR     ochar
        LDAA    #'F'
        JSR     ochar
        CLRA
        LDAB    RFA2    ;RIGHT
        JSR     HXTOA   ;INPUT IN D
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'L'
        JSR     ochar
        LDAA    #'F'
```

```
        JSR     ochar
        CLRA
        LDAB    LFA3    ;LEFT
        JSR     HXTOA   ;INPUT IN D
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'R'
        JSR     ochar
        LDAA    #'R'
        JSR     ochar
        CLRA
        LDAB    RRA6
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'L'
        JSR     ochar
        LDAA    #'R'
        JSR     ochar
        CLRA
        LDAB    LRA7
        JSR     HXTOA
        LDX     #CRLF
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'I'
        JSR     ochar
        LDAA    #'1'
        JSR     ochar
        LDAA    #'_'
        JSR     ochar
        CLRA
        LDAB    SI1A8
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'I'
        JSR     ochar
        LDAA    #'2'
        JSR     ochar
        LDAA    #'_'
        JSR     ochar
        CLRA
        LDAB    SI2A4
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'I'
        JSR     ochar
        LDAA    #'3'
        JSR     ochar
        LDAA    #'_'
        JSR     ochar
        CLRA
        LDAB    SI3A5
        JSR     HXTOA
        LDX     #CRLF
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
```

```
        LDAA    #'I'
        JSR     ochar
        LDAA    #'1'
        JSR     ochar
        LDAA    #'M'
        JSR     ochar
        CLRA
        LDAB    SI1MIN
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'I'
        JSR     ochar
        LDAA    #'2'
        JSR     ochar
        LDAA    #'M'
        JSR     ochar
        CLRA
        LDAB    SI2MIN
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'I'
        JSR     ochar
        LDAA    #'3'
        JSR     ochar
        LDAA    #'M'
        JSR     ochar
        CLRA
        LDAB    SI3MIN
        JSR     HXTOA
        LDX     #CRLF
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'1'
        JSR     ochar
        CLRA
        LDAB    S1
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'2'
        JSR     ochar
        CLRA
        LDAB    S2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'S'
        JSR     ochar
        LDAA    #'3'
        JSR     ochar
        CLRA
        LDAB    S3
        JSR     HXTOA
        LDX     #CRLF
        JSR     PrintS
****************************************
        LDAA    #'M'
```

```
        JSR     ochar
        LDAA    #'L'
        JSR     ochar
        CLRA
        LDAB    MIDL    ;LEFT CENTER
        JSR     HXTOA   ;INPUT IN D
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'M'
        JSR     ochar
        LDAA    #'R'
        JSR     ochar
        CLRA
        LDAB    MIDR    ;RIGHT CENTER
        JSR     HXTOA   ;INPUT IN D
        LDX     #CRLF
        JSR     PrintS
****************************************
        RTS

SEGMENT6        NOP
        LDAA    #'A'
        JSR     ochar
        LDAA    #'L'
        JSR     ochar
        CLRA
        LDAB    AVDL    ;AVOID LEFT CONTROL
        JSR     HXTOA   ;INPUT IN D
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'A'
        JSR     ochar
        LDAA    #'R'
        JSR     ochar
        CLRA
        LDAB    AVDR    ;AVOID RIGHT CONTROL
        JSR     HXTOA   ;INPUT IN D
        LDX     #COL
        JSR     PrintS
****************************************
        LDAA    #'R'
        JSR     ochar
        LDAA    #'M'
        JSR     ochar
        LDAA    #'T'
        JSR     ochar
        CLRA
        LDAB    REMOTE  ;REMOTE ON/OFF
        JSR     HXTOA   ;INPUT IN D

****************************************
        LDX     #CRLF
        JSR     PrintS
        RTS


SEGMENT7        JSR     PRNTPWA2
        LDX     #POS
        JSR     PrintS

****************************************
        RTS


*----------------------------------------
*       Print PWA2 Data
*----------------------------------------

PRNTPWA2        LDD     PWA2
```

```
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     LA2
        JSR     HXTOA
        RTS

*---------------------------------------
*       Print COLON
*---------------------------------------

PRNTCOLON       LDX     #COL
        JSR     PrintS
        RTS


*---------------------------------------
*       Print CRLF
*---------------------------------------
PRNTCRLF        LDX     #CRLF
        JSR     PrintS
        RTS

*---------------------------------------
*       Print OC2-3 VALUES
*---------------------------------------

PRNTOC23        NOP
        LDD     OC2LTH
        SUBD    OC2HTL
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     LTHDIFF2
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS

        LDD     OC3LTH
        SUBD    OC3HTL
        JSR     HXTOA
        LDX     #COL
        JSR     PrintS
        LDD     LTHDIFF3
        JSR     HXTOA
        RTS


*-----------------------
* DELAY FUNCTIONS
*-----------------------

PRNTPOS LDX     #POS
        JSR     PrintS
        RTS


*-----------------------
* DELAY FUNCTIONS
*-----------------------

XDELAY  PSHY
        LDY     T
        STX     T
        JSR     DELAY
        STY     T       ;RESTORE ORIGINAL T
        PULY
        RTS

DELAY   PSHA
        PSHB
```

```
       LDD    #0
DCOUNT ADDD   #1      ;14 cycles x factor in D
       CPD    T       ;before call to delay
       BNE    DCOUNT
       PULB
       PULA
       RTS

LDELAY JSR    DELAY
       JSR    DELAY
       JSR    DELAY
       JSR    DELAY
       RTS


************************************************************
*
*       INITS
*
************************************************************


*-----------------------
* INIT SONAR SYSTEM
*-----------------------

INIT_SONAR     LDAA   SI1A8
       ADDA   #2
       STAA   SI1MIN
       LDAA   SI2A4
       ADDA   #2
       STAA   SI2MIN
       LDAA   SI3A5
       ADDA   #2
       STAA   SI3MIN
       NOP
       RTS


*-----------------------
* INITIALIZE IR SYSTEM
*-----------------------

INIT_IR LDD    #$ff
       STD    ME11IO
       LDAA   LFA3
       ADDA   #TOLLERANCE
       STAA   MIDL
       LDAA   RFA2
       ADDA   #TOLLERANCE
       STAA   MIDR
       RTS


*-----------------------
* INITIALIZE SERVOS
*-----------------------

INIT_SERVOS    PSHA
       PSHB
       LDD    #1      ;init steering factors
       STD    STEER
       STD    ACCEL   ;init acceleration factors
       LDD    #100
       STD    ACCELF
       LDD    #4
       STD    DECELF
       LDD    #$FFFF
       STD    T
       LDD    #$FF
       STD    TURN    ;Init. turn time factor
```

```
        JSR     PW1INIT ;Init. Steering pulse to center
        JSR     PW2INIT ;Init. Speed Control pulse to halt
        PULB
        PULA
        RTS


*-----------------------
* INITIALIZE COUNTER
*-----------------------

INIT_COUNT      PSHA
        PSHB
        LDX     #BASE
        LDD     #0
        STD     TCNT,X          ;RESET COUNTER
        PULB
        PULA
        RTS


*-----------------------
* PULSE INITS
*-----------------------

* Initialize OCx pulses

PW1INIT LDX     #BASE
        LDD     PW1
        STD     TOC2,X  ;first transition Hi->Lo
        STD     CENTER1 ;CENTER
        RTS


PW2INIT LDX     #BASE
        LDD     PW2
        STD     TOC3,X  ;TEST CODE
        STD     CENTER2 ;HALT
        ADDD    #SLOWR  ;INIT. SLOW MOTION
        STD     SLOWREV
        LDD     CENTER2
        SUBD    #SLOWF
        STD     SLOWFWD
        RTS


DUMMYC  LDD     #DUMMYP
        STD     PW1
        STD     PW2
        STD     CENTER1
        STD     CENTER2
        LDD     #DUMMYL
        STD     L1
        STD     L2
        LDD     #DUMMYP
        STD     PWA1
        STD     PWA2
        LDD     #DUMMYL
        STD     LA1
        STD     LA2
        LDD     #DUMMYMIN
        STD     MIN1
        STD     MIN2
        LDD     #DUMMYMAX
        STD     MAX1
        STD     MAX2
        RTS


*-----------------------
* TIMER INIT
```

```
*-----------------------

* Set the Pre-Scaler Frequency for TCNT--first 64 cycles

TIMER   LDX     #BASE
        BCLR    TMSK2,X BIT1   ;PR1:PRO=00 FOR 32.768 msec
        BCLR    TMSK2,X BIT0   ;New count every 500ns
        RTS


*-----------------------
* IC1-2 INIT
*-----------------------

IC12INIT        LDX     #BASE
        BSET    TCTL2,X BIT2   ;EDG2B:EDG2A=01
        BCLR    TCTL2,X BIT3   ;CAPTURE RISING EDGE FIRST
        BSET    TCTL2,X BIT4   ;EDG1B:EDG1A=01
        BCLR    TCTL2,X BIT5   ;CAPTURE RISING EDGE FIRST
        BSET    TMSK1,X BIT1   ;ENABLE IC2 INTERRUPT
        BSET    TMSK1,X BIT2   ;ENABLE IC1 INTERRUPT
        LDD     #0
        STD     MAX1           ;INITIALIZE MAX/MIN
        STD     MAX2           ;CALCULATIONS FOR IC
        LDD     #$FFFF
        STD     MIN1
        STD     MIN2
        RTS


*-----------------------
* OC2-3 INITS (HI=11, LO=10)
*-----------------------

* INITIALIZE OC2 (Bit7:6=OM2:OL2)

OC2INIT LDX     #BASE
        BSET    TCTL1,X BIT7   ;OM2:OL2=11
        BSET    TCTL1,X BIT6   ;FOR SET TO HIGH
        BSET    TMSK1,X BIT6   ;ENABLE OC2
        RTS                    ;  INTERRUPT (Pulse 1)

* INITIALIZE OC3 (Bit5:4=OM3:OL3)

OC3INIT LDX     #BASE
        BSET    TCTL1,X BIT5   ;OM3:OL3=11
        BSET    TCTL1,X BIT4   ;FOR SET TO HIGH
        BSET    TMSK1,X BIT5   ;ENABLE OC3
        RTS                    ;  INTERRUPT (Pulse 2)


*-----------------------
* A/D INIT
*-----------------------

ADINIT  PSHA
        LDX     #BASE
        BSET    OPTION,X BIT7  ;TURN A/D ON
        LDAA    #40
WAIT    DECA
        BNE     WAIT           ;200 ECYCLES FOR STABILITY

        PULA
        RTS


*-----------------------
* RTI INIT
*-----------------------

INITRTI BSET    PACTL,X BIT0   ;8.192ms RTI
```

```
        BCLR    PACTL,X BIT1
        BSET    TMSK2,X BIT6   ;Enable RTI interrupts
        RTS

*********************************************************
*
* INTERRUPT SERVICE SUBROUTINES
*
*********************************************************



*-------------------------------------------------
* INPUT COMPARE 1
*-------------------------------------------------
IC1ISR  LDX     #BASE
        BRCLR   TFLG1,X BIT2 IC1EXIT  ;BRANCH AT ILLEGAL
        BCLR    TFLG1,X INV2   ;CLEAR FLAG
        BRCLR   TCTL2,X BIT4 IC1TOLO
IC1TOHI BSET    TCTL2,X BIT5   ;EDG1B:EDG1A=10
        BCLR    TCTL2,X BIT4   ;CAPTURE FALLING NEXT
        LDD     TIC1,X
        STD     TOHI1          ;PERIOD1 = CURRENT LO TO HI
        SUBD    PTOHI1         ; TRANSITION -
        STD     P1             ; PREVIOUS LO TO HI TRANS.
        BRA     IC1EXIT
IC1TOLO BCLR    TCTL2,X BIT5   ;EDG1B:EDG1A=01
        BSET    TCTL2,X BIT4   ;CAPTURE RISING NEXT
        LDD     TIC1,X
        STD     TOLO1
        SUBD    TOHI1          ;PW1 = TOLO1 - PTOHI1
        STD     PW1

        JSR     RMTECHK1              ;DETECT PULSE WIDTH CHANGE

        LDD     TOHI1
        STD     PTOHI1         ;UPDATE PREVIOUS TO CURRENT
        LDD     P1             ;L1 = P1 - PW1
        SUBD    PW1
        STD     L1
IC1EXIT NOP
*-----------------------------------------
*       USE OF IC1 FOR A/D CHECKING
*       AND NIMMAX CALCULATIONS
        JSR     CHECK
        JSR     MINMAX1
        JSR     MINMAX2
*-----------------------------------------
        RTI

RMTECHK1        LDD     CENTER1
        SUBD    PW1
        BLT     TOPOS1
RCHKCMP1        CPD     #THRESHOLD
        BLO     NORMT1
        LDAA    #1
        STAA    REMOTE1
        BRA     RCHKEXIT1
TOPOS1  STD     PPW1
        LDD     #0
        SUBD    PPW1
        BRA     RCHKCMP1
NORMT1  CLR     REMOTE1
RCHKEXIT1       JSR     REMCHK
        RTS



*-------------------------------------------------
* INPUT COMPARE 2
*-------------------------------------------------
```

```
IC2ISR  LDX     #BASE
        BRCLR   TFLG1,X BIT1 IC2EXIT  ;BRANCH AT ILLEGAL
        BCLR    TFLG1,X INV1   ;CLEAR FLAG
        BRCLR   TCTL2,X BIT2 IC2TOLO
IC2TOHI BSET    TCTL2,X BIT3   ;EDG2B:EDG2A=10
        BCLR    TCTL2,X BIT2   ;CAPTURE FALLING NEXT
        LDD     TIC2,X
        STD     TOHI2          ;PERIOD2 = CURRENT LO TO HI
        SUBD    PTOHI2         ; TRANSITION -
        STD     P2             ; PREVIOUS LO TO HI TRANS.
        BRA     IC2EXIT
IC2TOLO BCLR    TCTL2,X BIT3   ;EDG2B:EDG2A=01
        BSET    TCTL2,X BIT2   ;CAPTURE RISING NEXT
        LDD     TIC2,X
        STD     TOLO2
        SUBD    TOHI2          ;PW2 = TOLO2 - PTOHI2
        STD     PW2

        JSR     RMTECHK2

        LDD     TOHI2
        STD     PTOHI2         ;UPDATE PREVIOUS TO CURRENT
        LDD     P2             ;L2 = P2 - PW2
        SUBD    PW2
        STD     L2
IC2EXIT NOP
*----------------------------------------
*       USE OF IC2 FOR A/D SAMPLING
        JSR     ADNOI
*----------------------------------------
        RTI

RMTECHK2        LDD     CENTER2
        SUBD    PW2
        BLT     TOPOS2
RCHKCMP2        CPD     #THRESHOLD
        BLO     NORMT2
        LDAA    #1
        STAA    REMOTE2
        BRA     RCHKEXIT2
TOPOS2  STD     PPW2
        LDD     #0
        SUBD    PPW2
        BRA     RCHKCMP2
NORMT2  CLR     REMOTE2
RCHKEXIT2       JSR     REMCHK
        RTS


*--------------------------------------------------
* REMOTE CONTROL CHECK
*--------------------------------------------------

REMCHK  LDAA    REMOTE1
        ADDA    REMOTE2
        CMPA    #0
        BHI     REMOTE_ON
        CLRA
        STAA    REMOTE
        BRA     REMDONE
REMOTE_ON       LDAA    #1
        STAA    REMOTE
REMDONE RTS


*--------------------------------------------------
* OUTPUT COMPARE 2
*--------------------------------------------------

OC2IS   LDX     #BASE
        BRCLR   TFLG1,X BIT6 EXITOC2  ;IGNORE ILLEGAL INTERRUPT
```

```
        BCLR    TFLG1,X INV6   ;Clear flag
        BRSET   TCTL1,X BIT6 TOLOW2   ;End of positive pulse part
        BSET    TCTL1,X BIT7   ;SET TO HIGH: OM2:OL2=11
        BSET    TCTL1,X BIT6
        LDD     TOC2,X         ;LOAD CURRENT OC VALUE
        SUBD    OC2LTH         ;SUBTRACT PREVIOUS LO TO HI
        STD     LTHDIFF2              ;STORE DIFFERENCE IN TEMP
        LDD     TOC2,X         ;STORE NEW LO TO HI
        STD     OC2LTH

        JSR     OC2CTRL1

        BRA     EXITOC2
TOLOW2  BSET    TCTL1,X BIT7   ;SET TO LOW: OM2:OL2=10
        BCLR    TCTL1,X BIT6
        LDD     TOC2,X         ;LOAD CURRENT HI TO LO
        STD     OC2HTL         ;STORE IN TEMP

        JSR     OC2CTRL2

EXITOC2 RTI

OC2CTRL1        LDAA    REMOTE
        BEQ     AUTO21
        LDD     OC2LTH
        ADDD    L1
        BRA     CTRL21
AUTO21  LDD     OC2LTH
        ADDD    LA1
CTRL21  STD     TOC2,X         ;SET NEXT (PERIOD BEGINS)
        RTS

OC2CTRL2        LDAA    REMOTE
        BEQ     AUTO22
        LDD     OC2HTL
        ADDD    PW1
        BRA     CTRL22
AUTO22  LDD     OC2HTL
        ADDD    PWA1
CTRL22  STD     TOC2,X
        RTS



*-------------------------------------------------
* OUTPUT COMPARE 3
*-------------------------------------------------

OC3IS   LDX     #BASE
        BRCLR   TFLG1,X BIT5 EXITOC3  ;IGNORE ILLEGAL INTERRUPT
        BCLR    TFLG1,X INV5   ;Clear flag
        BRSET   TCTL1,X BIT4 TOLOW3   ;End of positive pulse part
        BSET    TCTL1,X BIT5   ;FOR SET TO HIGH: OM3:OL3=11
        BSET    TCTL1,X BIT4
        LDD     TOC3,X
        SUBD    OC3LTH         ;D = CURRENT LO TO HI - OC3LTH
        STD     LTHDIFF3              ;STORE DIFFERENCE IN TEMP
        LDD     TOC3,X         ;LOAD CURRENT
        STD     OC3LTH         ;STORE AS PREVIOUS

        JSR     OC3CTRL1

        BRA     EXITOC3
TOLOW3  BSET    TCTL1,X BIT5   ;FOR SET TO LOW: OM3:OL3=10
        BCLR    TCTL1,X BIT4
        LDD     TOC3,X
        STD     OC3HTL

        JSR     OC3CTRL2

EXITOC3 RTI
```

```
OC3CTRL1       LDAA   REMOTE
       BEQ     AUTO31
       LDD     OC3LTH
       ADDD    L2
       BRA     CTRL31
AUTO31 LDD     OC3LTH
       ADDD    LA2
CTRL31 STD     TOC3,X         ;SET NEXT (PERIOD BEGINS)
       RTS

OC3CTRL2       LDAA   REMOTE
       BEQ     AUTO32
       LDD     OC3HTL
       ADDD    PW2
       BRA     CTRL32
AUTO32 LDD     OC3HTL
       ADDD    PWA2
CTRL32 STD     TOC3,X
       RTS


*---------------------------------------
*      A/D SERVICE/NO INTERRUPTS
*---------------------------------------

ADNOI  NOP
       LDX     #BASE
       LDAA    #%00010000
       STAA    ADCTL,X

CONV1_4 BRCLR  ADCTL,X BIT7 CONV1_4  ;Wait until flag is set

       LDAA    ADR2,X
       STAA    RFA2
       LDAA    ADR3,X
       STAA    LFA3
       LDAA    ADR4,X
       STAA    SI2A4

       LDAA    #%00010100
       STAA    ADCTL,X

CONV5_8 BRCLR  ADCTL,X BIT7 CONV5_8  ;Wait until flag is set

       LDAA    ADR5,X
       STAA    SI3A5
       LDAA    ADR6,X
       STAA    RRA6
       LDAA    ADR7,X
       STAA    LRA7
       LDAA    ADR8,X
       STAA    SI1A8

ACONT2 NOP
       JSR     SMINS
       JSR     SDIFF

       RTS


*---------------------------------------
*      ichar
*---------------------------------------

ichar  LDAA    SCSR    ; Check status reg.
*                      ;     (load it into A reg)
       ANDA    #$20    ; Check if receive buffer full
       BEQ     ichar   ; Wait until data present
       LDAA    SCDR    ; SCI data ==> A register
```

```
        RTS             ; Return from subroutine


*---------------------------------------
*       ochar; Input in A.
*---------------------------------------

ochar   PSHB            ; Save contents of B register
Loop1   LDAB    SCSR    ; Check status reg (load it into B reg)
        ANDB    #$80    ; Check if transmit buffer is empty
        BEQ     Loop1   ; Wait until empty
        STAA    SCDR    ; Register A ==> SCI data
        PULB            ; Restore B register
        RTS             ; Return from subtoutine


*---------------------------------------
*       PrintS; Index x[0]=first char.
*---------------------------------------

PrintS  PSHA            ; Save contents of A register
*       SEI             ; Disable interrupts
Loop2   LDAA    0,X         ; Get a character (put in A register)
        CMPA    #EOS        ; Check if it's EOS
        BEQ     Done        ; Branch to Done if it's EOS
        JSR     ochar   ; Print the character by calling ochar
        INX             ; Increment index
        BRA     Loop2   ; Branch to Loop2 for the next char.
Done    NOP             ; Dummy line
*       CLI             ; Enable interrupts;
        PULA            ; Restore A register;
        RTS             ; Return from subtoutine;


*---------------------------------------
*       CLEAR SCREEN
*---------------------------------------

CLRS    LDX     #CLR
        JSR     PrintS
        LDX     #POS
        JSR     PrintS
        RTS


*---------------------------------------
*       PRINT HEX IN ASCII (IN D)
*---------------------------------------

HXTOA   STD     WORD    ;STORE NO. IN TEMP
        ANDA    #%11110000      ;AND A WITH MASK
        LSRA
        LSRA
        LSRA
        LSRA
        JSR     ADJUST
        JSR     ochar
        LDD     WORD
        ANDA    #%00001111
        JSR     ADJUST
        JSR     ochar
        ANDB    #%11110000      ;AND B WITH MASK
        LSRB
        LSRB
        LSRB
        LSRB
        TBA
        JSR     ADJUST
        JSR     ochar
        LDD     WORD
        ANDB    #%00001111
```

```
        TBA
        JSR     ADJUST
        JSR     ochar
        RTS


ADJUST  INCA            ;ADJUST A FIRST TO ACC. FOR DECA
        LDX     #ZERO-1 ;ZERO TABLE INDEX FACTOR IN A
NUM     INX
        DECA
        BNE     NUM
        LDAA    0,X
        RTS


*---------------------------------------
*       HEX TO DEC CONVERSION
*       Index of input's address in X
*---------------------------------------

Convert JSR     HTOD            ;Convert to dec -> ASCII
        LDX     #DBUFR          ;Load converted no's index
        CLRB
Digits  LDAA    0,X             ;Load byte X [0:4]
        JSR     ochar           ;Output Byte in A
        INX                     ;X = X+1, next digit
        INCB                    ;Increment B
        CMPB    #5              ;Compare with total
        BNE     Digits          ;Branch if != 5
        RTS


*---------------------------------------
*       SCI Initialization
*---------------------------------------

InitSCI         PSHA            ; Save contents of A register
        LDAA    #$30    ; Set BAUD rate to 9600
        STAA    BAUD
        CLR     SCCR1   ; Set SCI Mode to 1 start bit,
*                       ;     8 data bits, and 1 stop bit.
        LDAA    #$0C    ; Enable SCI Transmitter
        STAA    SCCR2   ;     and Receiver
        PULA            ; Restore A register
        RTS             ; Return from subtoutine


*-----------------------------------------------
* PORTD INIT
*-----------------------------------------------

PDINIT  LDX     #BASE
        LDAA    #IOPAT
        STAA    DDRD,X
        RTS



*****************************************************************************
* HTOD - Subroutine to convert a 16-bit hex number to a
*         5 digit decimal number.
*
* Uses 5 byte variable "DBUFR" for decimal ASCII result
* On entry X points to hex value to be converted & displayed
* All registers are unchanged upon return
*****************************************************************************

HTOD    PSHX            ;Save registers
        PSHB
        PSHA
        LDD  0,X                ;D=hex value to be converted
```

```
        LDX  #10000
        IDIV            ;freq+10,000 -> X; r -> D
        XGDX            ;Save r in X; 10,000s digit in D (A:B)
        ADDB #$30               ;Convert to ASCII
        STAB DBUFR              ;Store in decimal buffer
        XGDX            ;r back to D
        LDX  #1000
        IDIV            ;r+1,000 -> X; r -> D
        XGDX            ;Save r in X; 1,000s digit in D (A:B)
        ADDB #$30               ;Convert to ASCII
        STAB DBUFR+1    ;Store in decimal buffer
        XGDX            ;r back to D
        LDX  #100
        IDIV            ;r+100 -> X; r -> D
        XGDX            ;Save r in X; 100s digit in D (A:B)
        ADDB #$30               ;Convert to ASCII
        STAB DBUFR+2    ;Store in decimal buffer
        XGDX            ;r back to D
        LDX  #10
        IDIV            ;r+10 -> X; r in D (B is units digit)
        ADDB #$30               ;Convert to ASCII
        STAB DBUFR+4    ;Store to units digit
        XGDX            ;10s digit to D (A:B)
        ADDB #$30               ;Convert to ASCII
        STAB DBUFR+3    ;Store in decimal buffer
        PULA            ;Restore registers
        PULB
        PULX
        RTS             ;Return

*******************************************************************************

        END
```