Georgia Tech

THE GEORGE W. WOODRUFF SCHOOL OF
MECHANICAL ENGINEERING

Georgia Institute of Technology
Atlanta, Georgia 30332-0405

ME 8501

Mechanical Design Special Topics

Nasa/University

Advanced Design Program

Proof of Principle Model for a Lunar Dumptruck

June 1989

Jill Ann Harvey

Problem Statement

Background

It was desired to design a proof-of-principle model of a two wheeled lunar dump truck, to display at the 1989 Summer Conference in Huntsville, Al. The dump truck, which was designed fall quarter, consists of a cylindrical bucket suspended between two wheels with the center of gravity of the bucket lower than the axis of the wheels (ref. fig 1). This gives the truck great stability in rough terrain. One motor on each wheel controls the movement of the truck and limits the moving parts to only two.

Requirements

It is desired in the future, to use the curvilinear motors,currently under design at the V.A. hospital in Atlanta, Ga., to power the lunar version of the truck. However, for the proof-of-principle model, the motors and drive system of an Everest and Jennings wheel chair were modified to meet the minimum requirements. The truck needed to be operated from a D.C. source, and be remotely controlled. Also, to facilitate dumping, all parts must be contained within the wheel base.

Constraints

As previously mentioned, a wheelchair was to be modified for the truck. Therefore, the motors that were used were 24 Volt D.C. motors. Also, in order for the truck to be able to invert, the driver pulley had to be relocated within the wheelbase. The belt length had to remain constant as the poly-rib belt used on the chair was not available in a variety of lengths. Finally, the truck needed to remain as safe as possible; the batteries had to bolted in the bowl to minimize the chance acid spill during dumping.

Mechanical Drive Design

The mechanical drive consists of three separate parts for each side of the dump truck. The driver pulley is powered by the motor. The driven pulley is attached to the wheel and pulled be a k section poly-rib belt. Because of the necessity that the wheelchair must be reassembled, none of the parts could be altered. There were basically three possible methods to construct the system: shortening the span, adding a double pulley, or using a mule drive. Shortening the span would be the easiest method of attachment. However, the tolerancing would be very tight, and it is difficult to find poly-rib belting in variable lengths. Using a double pulley with a slightly smaller diameter than the driver pulley would help the tolerancing problem but, adding and extra pulley and another belt became too expensive. The mule drive, while not desirable, satisfied the necessary requirements for short term operation.

A mule drive consists of four pulleys arranged so the belt makes a 90 degree turn (see fig 2). It was decided that pulleys 2 and 4 would be .5 in. diameter McGill bearings. These bearings will also be used to tension the belt after the driver pulley is mounted in place. The motors will be mounted below the axis of the wheels to help keep the center of gravity low. The relatively short length of the belt predetermined the placement of the pulleys to attain the most wrap in all directions. It is necessary to attain at least 50% wrap on both the driver and driven pulleys. By placing the idler bearings relatively close to the wheel pulley, about 270 degrees of wrap can be obtained. After compensating for the belt length, this leaves approximate spans of around 6 inches to the driver pulley. After adjusting this pulley for proper tensioning, the values for the angular placement of the idlers can be determined.

Electrical System

An electrical system to connect the motors to the microcontroller was needed. The circuit needed to be isolated from the motor to guard against current spikes. Also, the circuit needed to be responsive to pulse width modulated input signals.

Generally, circuit design at this level of electronics consists of simple relays, resistors, and transistors. The circuit outlined in Fig. 3 will drive a motor with relatively low current needs. All inputs are active low, and are configured for TTL compatible signals. The logic source would be connected to the same 5 Volt source connected to the processor. The transistors provide the current gain needed to energize one of the relays. The wire across the base of the motors allows for the pulse width modulated signal. The diodes are simply protection from current backflow after the circuit is de-energized. The circuit is fully designed for a 12 or 6 Volt motor which would pull less than 10 amps. of stall current.

Unfortunately, the motors on the wheelchair had no specifications. A few simple measurements were made to determine the current requirements of the motor. The starting current was around 40 amps. After gathering data from similar motors, the stall current was estimated to be somewhere between 60 and 100 amps.

This is far too much current for the simple electrical system shown above. In order to obtain a relay capable of handling just the start-up current, a double pole relay would have to be wired across the connection plates.

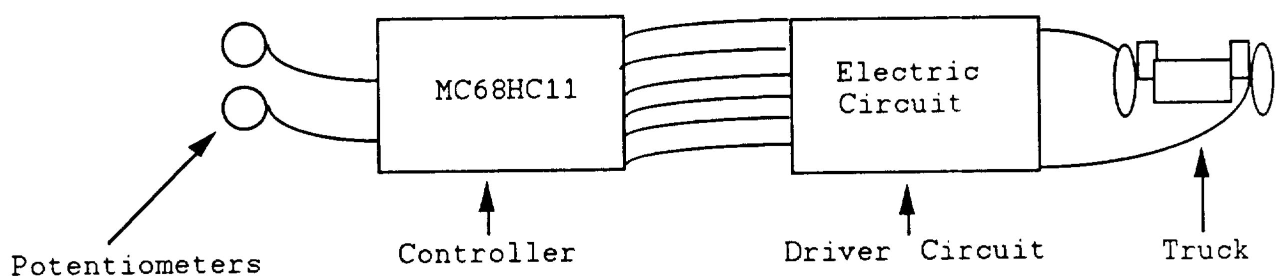
Included with the electrical system are two ten turn 10k ohm potentiometers. These provide the input signal to the microprocessor for speed and direction control. They will also be powered to 5 VDC and wired to the A/D converter located on the processor (see fig below). This will provide the control system with 255 different levels of direction and speed control.

Control System

The objective of the control system is to coordinate the control of the two motors via a remote control station (i.e. self contained). It was decided that digital control would provide the simplest method of integrating the desired input with the output signals. After careful investigation, the Motorola MC68HC11 was chosen as the microcontroller. The HC11 provided a 7 channel A/D controller onboard in addition to 5 output compare registers for pulse width modulated output. Also located on the chip, are 512 bytes of EEPROM memory for permanent memory storage so the controller can be disconnected from the computer.

The input signals are provided by the two potentiometers. After A/D conversion, the variable voltage of the potentiometer is converted to an 8 byte hexadecimel number between 00 and FF. The median value of input is HEX 7F. This corresponds to the division between forward and reverse on the speed control or left and right on the direction control. The machine code program reads the values from the A/D registers then determines which control needs to be taken.

The output signals are generated through the output compare registers. These registers are loaded with a HEX number between 0000 and FFFF. These numbers correspond to the free running clock on the chip. When the clock reaches the number stored in the register, a specified action is taken (the line either goes high or low). This generates a waveform which in turn drives the motor. The waveform actually turns the motor off and on but, it happens so quickly that it is impossible to notice. Since the electric circuit in the previous section was designed for active low signals, the register drives the output high when a match occurs. This stops the ontime of the motor.

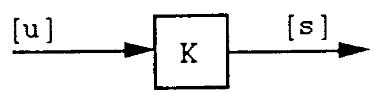


The correlation between the direction control and speed control is linear. Once the desired speed is determined, a direction offset is calculated and that amount is added to motors ontime and subtracted from the others. The input signals are given a "gain" of HEX 101 to convert from the FF scale to FFFF. This required some creative programming as the chip will not perform 16 bit multiplications. To compress the code further, the conversion process could be placed in a subroutine rather than repeated so often.

The state space equations of control are:

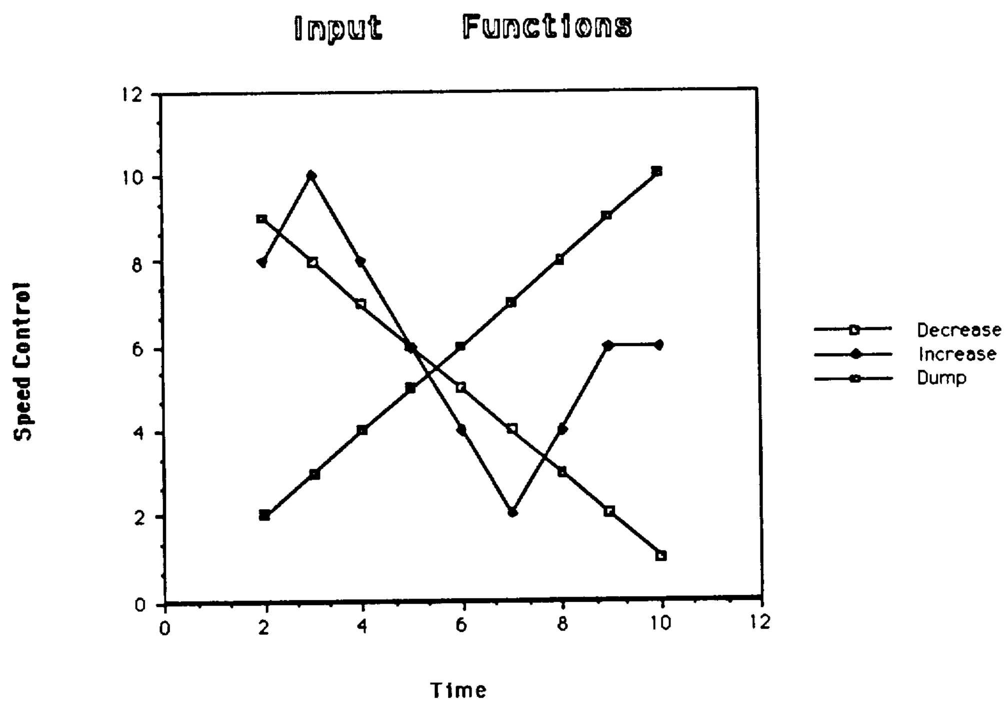
$$[\underline{S}] = K \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} [\underline{U}]$$

where the $[\underline{S}]$ matrix corresponds to the speeds of the motors; K is the gain HEX 101; and [U] is the input signals from the pots.



The program, as written, uses 284 cycles of processing time to calculate the output signals for the motors. This corresponds to a sampling time of 14.2 milliseconds. Relative to the slowness of response of the inertia of the mechanical system, this is extremely fast control. If a closed loop system were implemented, a time delay would have to be added to maintain system stability. The system would easily over calculate while waiting for the motor shaft to attain the desired speed. In the open loop system, the operator closes the loop by controlling the rate of change for input parameters.

The simple graph shown below demonstrates the input signals for desired speed control. The non-linear curve demonstrates the input necessary for the dumping action. By quickly advancing the pot then quickly decreasing it, the dumping action will occur. The other two are linear and correspond to slowing down or speeding up.



Conclusions and Recommendations

The mechanical drive is questionable. The poly rib belt is not designed to carry a load on the back side. Therefore, twisting the belt is against the desired operating conditions. However, for the relatively short duration of the demonstration, this would suffice.

The electrical circuit which connects the motors to the board and provides current protection for the board is designed properly for low power circuits. The 24 Volt motors require a heavy duty electric interface circuit. Therefore, the circuit as shown will not work.

The MC68HC11 provides excellent control for the lunar dump truck proof-of-principle model. This chip has all the functions necessary to implement the control on board. Also, the E clock rate of 500 nsec per cycle provides fast response time to input signals.

It is recommended that 12 Volt, 10 amp motors be used for the design or, an electrical engineer work on the interface circuit. For the short duration of the demonstration, the mule drive will give sufficient torque. However, if a more permanent model were to be built, a direct drive system would perform much better. Finally, to improve the reaction of the truck to obstacles, closed loop feedback control should be implemented.

Bibliography

Heiseman, David. How to Design and Build Your Own Custom Robot. Blue Ridge Summit, Pa. : TAB Books, 1981.

Motorola. Advance Information : MC68HC11A8 HCMOS Single-Chip Microcontroller. Phoenix, Az. : Motorola, 1988.

Motorola. M68HC11 HCMOS Single-Chip Microcontroller : Programmer's Reference Manual. Phoenix, Az. : Motorola.

Oliver, L.R., Johnson, C.O., and Breig, W.F.. Agricultural V-Belt Drive Design. Dayton, Oh.: Dayco Corp. 1977.

Peatman, John B.. Design with Microcontrollers. New York: McGraw-Hill Book Company, 1988.

Staugaard, Andrew. 6801, 68701, and 6803 Microcomputer Programming and Interfacing. Indianaplois, In. : Howard W. Sams & Co., Inc, 1980.

Waugh, Dale L., David G. Fisher, and Melvin D. Gayer, eds. Dayco Automotive Belt Drive Design Handbook. Springfield, Missouri, 1980

Appendix A

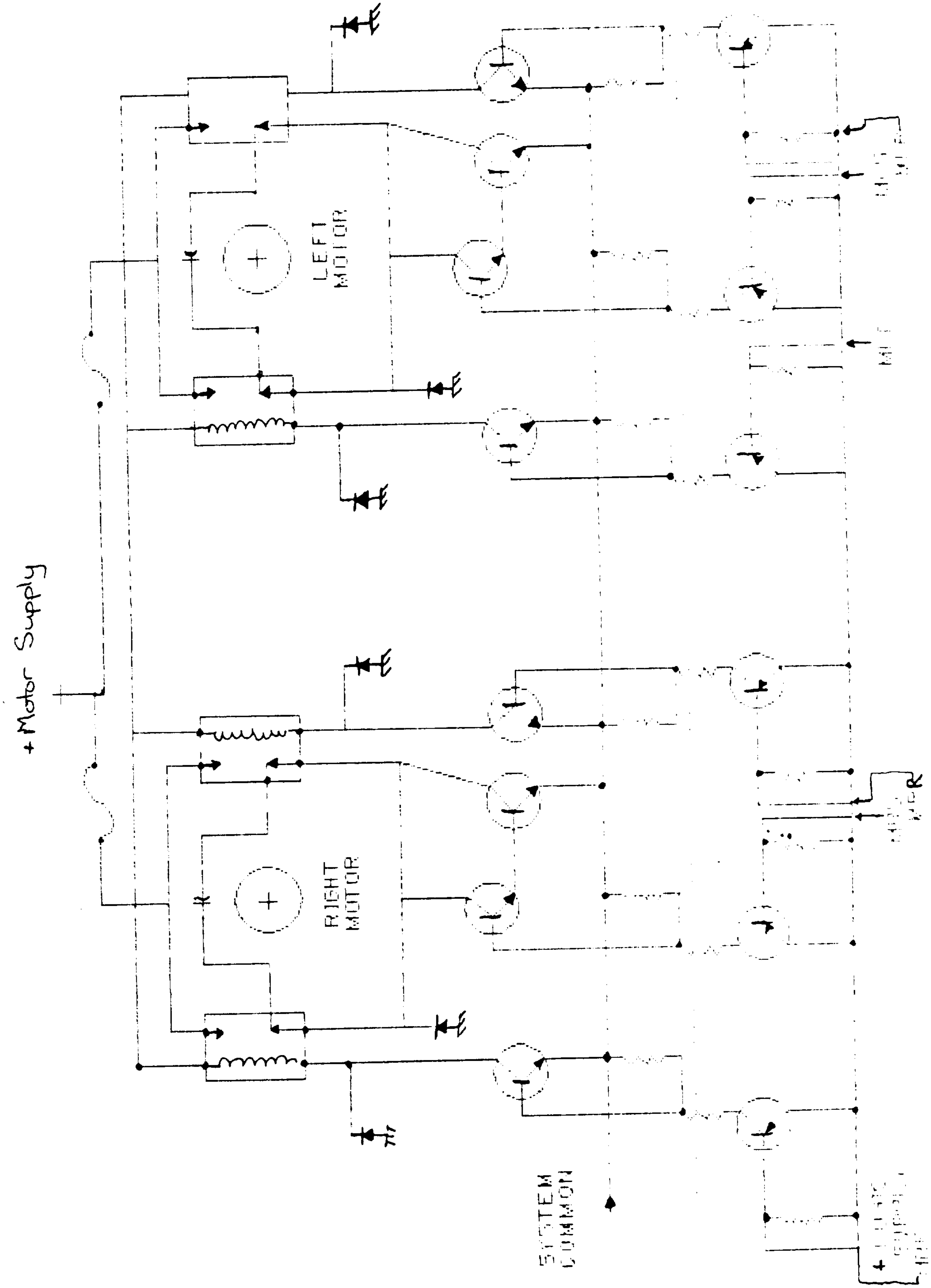
Electric Circuit Diagram

and

Controls Program

for

MC68HC11



ORIGINAL PAGE IS
OF POOR QUALITY

```
                  **************************************************
                  *                                                *
                  *      THIS PROGRAM WILL PROVIDE                 *
                  *      DIGITAL CONTROL FOR A SMALL               *
                  *      TWO WHEELED DUMP TRUCK.                   *
                  *      THE PROGRAM USES THE ONBOARD              *
                  *      A/D CONVERTER TO DETERMINE                *
                  *      DESIRED VALUES FOR SPEED AND              *
                  *      DIRECTION.  THE CORRESPONDING             *
                  *      OUPUT SIGNALS ARE THEN SENT               *
                  *      THROUGH PORT A OUTPUT LINES.              *
                  *      THE SAMPLING TIME OF THE                  *
                  *      CONTROLLER HAS BEEN DETERMINED            *
                  *      TO BE *****.                              *
                  *                                                *
                  **************************************************

000                    ORG $C000 : THIS STARTS THE PROGRAM AT
                  *              : MEMORY LOCATION C000

                  **************************************************
                  *           PARAMETER INITIALIZATION             *
                  **************************************************

000 86 90              LDAA #$90    :THIS POWERS UP THE
002 B7 10 39           STAA $1039   :A/D CONVERSIONS

005 86 CC              LDAA #$CC    :THIS CONFIGURES THE
007 B7 10 20           STAA $1020   :TIMER CNTL REG. FOR
                  *                 :OUTPUT SIGNALS

00A 86 0F              LDAA #$0F    :THIS CONFIGURES PORT C
00C B7 10 07           STAA $1007   :DIRECTION CONTROL SIGNALS
                  *                 :(PINS 0,1,2,3)

00F 86 02              LDAA #$02    :STORAGE OF CONVERSION
011 97 05              STAA $05     :CONSTANTS
013 86 FF              LDAA #$FF    :
015 97 06              STAA $06     :
                  *:CHECK IF YOU NEED TIME DELAY HERE JILL
                  **************************************************
                  *                START OF CONTROLS
                  **************************************************

017 86 10         TOP  LDAA #$10    :THIS CONFIGURES THE A/D
019 B7 10 30           STAA $1030   :REGISTER FOR MULTIPLE CHANNELS.
                  *                 :AND BEGINS CONVERSIONS (2&4)

01C B6 10 30      LP1  LDAA $1030   :CHECK FOR CONVERSIONS
                  *                 :COMPLETE FLAG  TO BE
01F 81 90              CMPA #$90    :SET. IF NOT KEEP WAITING
021 26 F9              BNE LP1      :

023 B6 10 32           LDAA $1032   :GATHER INPUT FROM DIRECTION
026 97 01              STAA $01
028 B6 10 34           LDAA $1034   :GATHER INPUT FROM SPEED
02B 97 02              STAA $02

                  **************************************************
```

```
                 *            BEGIN CALCULATIONS

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

)20 86 7F               LDAA ##7F   :THIS IS THE MEDIAN INPUT
)2F 91 02               CMPA $02    :TO DETERMINE THE DESIRED
)31 2F 07               BLE FWD     :DIRECTION

)33 86 0A               LDAA ##0A   :THIS OUTPUTS THE DIRECTION
)35 B7 10 03            STAA $1003  :COMMAND TO CICUIT
)38 20 09               BRA NXT
)3A 86 05         FWD   LDAA ##05
)3C B7 10 03            STAA $1003
)3F 20 02               BRA NXT
)41 20 D4         STP   BRA TOP
)43 96 02         NXT   LDAA $02    :CONVERSION OF INPUT
)45 D6 05               LDAB $05    :UNITS TO OUTPUT UNITS
)47 3D                  MUL         :OF ONTIME FOR MOTOR SPEED
)48 DD 10               STAD $10    :        "
)4A 96 02               LDAA $02    :        "
)4C D6 06               LDAB $06    :        "
)4E 3D                  MUL         :        "
)4F D3 10               ADDD $10    :        "
)51 DD 10               STAD $10    :STORE THE RESULT IN MEM

)53 86 7F               LDAA ##7F   :THIS IS THE MEDIAN FOR
)55 91 01               CMPA $01    :FOR STEERING CONTROL
)57 2F 24               BLE RT      :TO DETERMINE RIGHT & LEFT
)59 90 01         LFT   SUBA $01    :SUBTACT TO DETERMINE OFFSET
)5B 97 12               STAA $12    :STORE RESULT
)5D D6 05               LDAB $05    :BEGIN CONVERSION FROM INPUT
)5F 3D                  MUL         :UNITS TO OUTPUT UNITS
)60 DD 20               STAD $20    :        "
)62 96 12               LDAA $12    :        "
)64 D6 06               LDAB $06    :        "
 66 3D                  MUL         :        "
)67 D3 20               ADDD $20    :        "
 69 DD 20               STAD $20    :STORE THE RESULT IN MEM
)6B DC 10               LDAD $10
)6D 93 20               SUBD $20    :SUBTRACT OFFSET FROM
)6F DD 22               STAD $22    :DESIRED SPEED FOR LEFT MOTOR
)71 DC 10               LDAD $10    :ADD OFFSET TO
)73 D3 20               ADDD $20    :DESIRED SPEED FOR RIGHT MOTOR
)75 FD 10 1C            STAD $101C  :THIS SETS THE RIGHT MOTOR SPEED
)78 DC 22               LDAD $22    :
)7A FD 10 18            STAD $1018  :THIS SETS THE LEFT MOTOR SPEED
)7D 96 01         RT    LDAA $01    :THIS BEGINS THE RIGHT TURN
)7F 80 7F               SUBA ##7F   :
)81 97 20               STAA $20    :THIS IS THE OFFSET
)83 D6 05               LDAB $05    :BEGIN CONVERSION
)85 3D                  MUL         :        "
 86 DD 14               STAD $14    :        "
)88 96 20               LDAA $20    :        "
 8A D6 06               LDAB $06    :        "
)8C 3D                  MUL         :        "
 8D D3 14               ADDD $14    :        "
)8F DD 20               STAD $20    :THIS IS THE OFFSET
)91 DC 10               LDAD $10    :THIS SETS RIGHT MOTOR
)93 93 20               SUBD $20    :DESIRED SPEED FOR TURN
)95 DD 22               STAD $22    :
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
)97 0C 10             LDAD $10     :THIS SETS LEFT MOTOR
)99 D3 20             ADDD $20     :THIS IS LEFT MOTOR ONTIME

)9B FD 10 18          STAD $1018 :SET LEFT MOTOR TO SPEED
)9E DC 22             LDAD $22     :THIS IS RIGHT MOTOR ONTIME
)A0 FD 10 1C          STAD $101C :SET RIGHT MOTOR TO SPEED
)A3 20 9C             BRA STP      :RETURN TO CHECK SETTINGS AGAIN

rrors: 0
```

ORIGINAL PAGE IS
OF POOR QUALITY