Application 9: Interfacing a Stepper Motor to the PRIMER

Purpose: To show how a computer can be used to perform motion control using a stepper motor.

Goals:

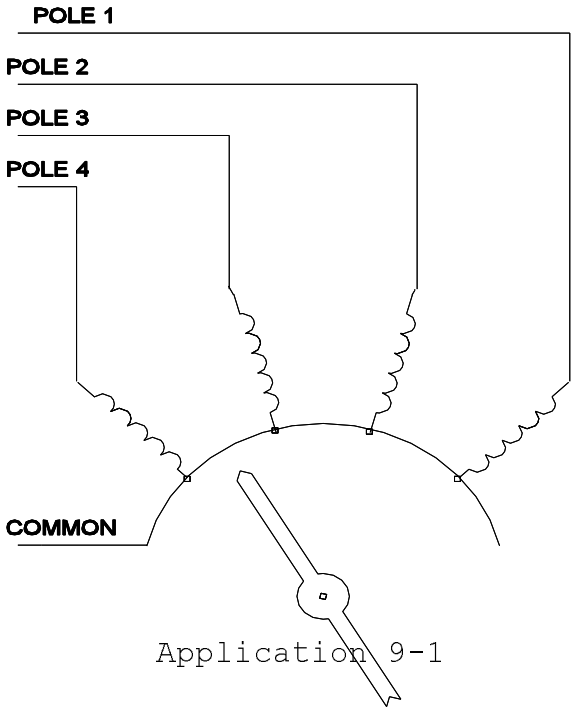
1. Build a stepper motor driver circuit.
2. Load a program that will demonstrate stepper motor control.

Materials:

1) PRIMER trainer
1) breadboard
1) SM4200 4 Phase stepper motor (Jameco part #105890. Call 1-800-831-4242)
1) 7404 Hex Inverter
4) 2N3904 NPN Transistors
4) 1N4001 Diodes
4) 1K Ohm, 1/4 Watt Resistor
1) 220 Ohm, 1/4 Watt Resistor

Discussion:

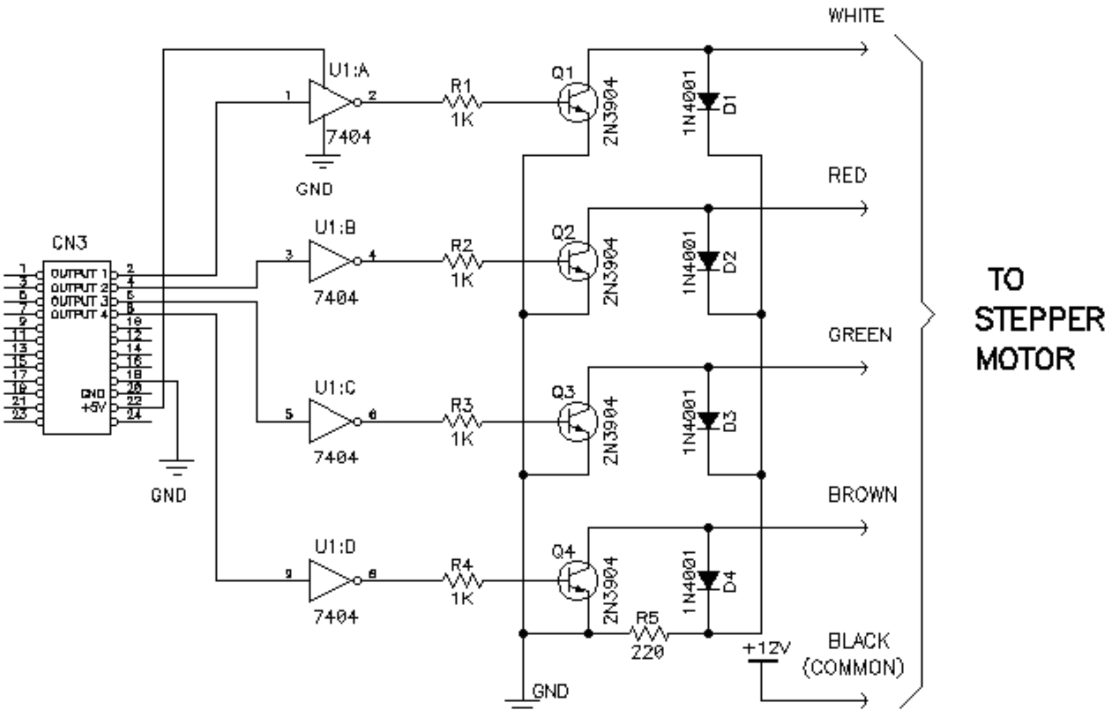
This lab shows how the PRIMER can be used to drive a stepper motor. The diagram below shows the electrical equivalent of a 4 phase stepper motor connected to the output port of the PRIMER. When the program first starts, OUTPUT2 and OUTPUT3 are energized. The stepper is now held in position because of the magnetic force pulling the rotor between the energized poles. A step can be made by turning on OUTPUT4 while turning off OUTPUT2. This moves the rotor one increment. To move one more increment, OUTPUT1 is turned on while OUTPUT3 is turned off. To go back to the original position, the sequence would be as follows: Turn on OUTPUT3 while turning off OUTPUT1, turn on OUTPUT2 while turning off OUTPUT4.



Application 9-1

Circuit Description and Construction:

The stepper motor cannot connect directly to the output port of the PRIMER because it uses 5 volt logic levels while the stepper motor operates on 12 volts. The current demand of the stepper motor is also a problem, since computer logic supplies very low current compared to the stepper motor's needs. The solution to these problems is an interface circuit. The circuit shown in the schematic provides the necessary interface from 5 volt logic to a 12 volt source required by the stepper. Transistors Q1-Q4 provide the current and voltage amplification while diodes D1-D4 and resistor R5 provide a feedback path for the back EMF generated when the poles are de-energized. The inverters are used to convert the negative logic on the PRIMER to positive logic and to prevent the stepper from being energized when the



PRIMER is reset. The interface is connected to the low nibble (4 bits) of the PRIMER output port. The driver circuit should be built on a breadboard following the schematic. Once built, a small piece of solid wire should be tightly wrapped around the shaft of the stepper motor to serve as a pointing device.

Note - The stepper motor and driver circuit are powered from a power supply separate from the PRIMER itself. This is necessary because of the large current draw and noise produce by the stepper motor.

Using the Program:

Load the following program into memory:

ADDRESS	DATA	INSTRUCTION	
FF01	1E	MVI	E,37
FF02	37		
FF03	16	MVI	D,01
FF04	01		
FF05	0E	MVI	C,11
FF06	11		
FF07	CD	CALL	1000
FF08	00		
FF09	10		
FF0A	1E	MVI	E,FB
FF0B	FB		
FF0C	15	DCR	D
FF0D	CD	CALL	1000
FF0E	00		
FF0F	10		
FF10	3E	MVI	A,33
FF11	33		
FF12	32	STA	FFAC
FF13	AC		
FF14	FF		
FF15	AF	XRA	A
FF16	32	STA	FFAD
FF17	AD		
FF18	FF		
FF19	6F	MOV	L,A
FF1A	47	MOV	B,A
FF1B	C3	JMP	FF43
FF1C	43		
FF1D	FF		
FF1E	78	MOV	A,B
FF1F	32	STA	FFAD
FF20	AD		
FF21	FF		
FF22	CD	CALL	FF4B
FF23	4B		
FF24	FF		
FF25	3A	LDA	FFAD
FF26	AD		
FF27	FF		
FF28	47	MOV	B,A
FF29	16	MVI	D,00
FF2A	00		
FF2B	58	MOV	E,B
FF2C	0E	MVI	C,13
FF2D	13		
FF2E	CD	CALL	1000
FF2F	00		
FF30	10		
FF31	7D	MOV	A,L
FF32	90	SUB	B
FF33	CA	JZ	FF1E
FF34	1E		
FF35	FF		

ADDRESS	DATA	INSTRUCTION	
FF36	DA	JC	FF3F
FF37	3F		
FF38	FF		
FF39	04	INR	B
FF3A	AF	XRA	A
FF3B	5F	MOV	E,A
FF3C	C3	JMP	FF43
FF3D	43		
FF3E	FF		
FF3F	05	DCR	B
FF40	AF	XRA	A
FF41	3C	INR	A
FF42	5F	MOV	E,A
FF43	16	MVI	D,64
FF44	64		
FF45	CD	CALL	FF72
FF46	72		
FF47	FF		
FF48	C3	JMP	FF29
FF49	26		
FF4A	FF		
FF4B	06	MVI	B,02
FF4C	02		
FF4D	0E	MVI	C,0B
FF4E	0B		
FF4F	CD	CALL	1000
FF50	00		
FF51	10		
FF52	7D	MOV	A,L
FF53	FE	CPI	0A
FF54	0A		
FF55	D2	JNC	FF4D
FF56	4D		
FF57	FF		
FF58	05	DCR	B
FF59	CA	JZ	FF62
FF5A	62		
FF5B	FF		
FF5C	32	STA	FFAA
FF5D	AA		
FF5E	FF		
FF5F	C3	JMP	FF4D
FF60	4D		
FF61	FF		
FF62	32	STA	FFAB
FF63	AB		
FF64	FF		
FF65	3A	LDA	FFAA

FF66	AA		
FF67	FF		
FF68	47	MOV	B,A
FF69	CD	CALL	FFA1
FF6A	A1		
FF6B	FF		

ADDRESS	DATA	INSTRUCTION	
FF6C	3A	LDA	FFAB
FF6D	AB		
FF6E	FF		
FF6F	80	ADD	B
FF70	6F	MOV	L,A
FF71	C9	RET	
FF72	F5	PUSH	PSW
FF73	C5	PUSH	B
FF74	7B	MOV	A,E
FF75	1F	RAR	
FF76	3A	LDA	FFAC
FF77	AC		
FF78	FF		
FF79	DA	JC	FF80
FF7A	80		
FF7B	FF		
FF7C	0F	RRC	
FF7D	C3	JMP	FF81
FF7E	81		
FF7F	FF		
FF80	07	RLC	
FF81	32	STA	FFAC
FF82	AC		
FF83	FF		
FF84	DB	IN	11
FF85	11		
FF86	E6	ANI	F0
FF87	F0		
FF88	47	MOV	B,A
FF89	3A	LDA	FFAC
FF8A	AC		
FF8B	FF		

ADDRESS	DATA	INSTRUCTION	
FF8C	E6	ANI	0F
FF8D	0F		
FF8E	B0	ORA	B
FF8F	D3	OUT	11
FF90	11		
FF91	D5	PUSH	D
FF92	06	MVI	B,FF
FF93	FF		
FF94	05	DCR	B
FF95	C2	JNZ	FF94
FF96	94		
FF97	FF		
FF98	00	NOP	
FF99	15	DCR	D
FF9A	C2	JNZ	FF92
FF9B	92		
FF9C	FF		
FF9D	D1	POP	D
FF9E	C1	POP	B
FF9F	F1	POP	PSW
FFA0	C9	RET	
FFA1	F5	PUSH	PSW
FFA2	78	MOV	A,B
FFA3	07	RLC	
FFA4	07	RLC	
FFA5	80	ADD	B
FFA6	07	RLC	
FFA7	47	MOV	B,A
FFA8	F1	POP	PSW
FFA9	C9	RET	

Once the program is started the LED display should read "0000 P0.". The "P0." Stands for "position" and "0000" indicates the relative position of the stepper referenced from its original position when the program was started (thus 0000 means it is in the same position as it was on start up). Press a two digit decimal number on the keypad and the stepper motor should move to that position with the display incrementing as the stepper moves. Once the stepper stops, enter 00 and the stepper should rotate the opposite direction with the display decrementing and finally stopping at 00. The stepper motor should now be in the exact position it was in when the program was first started.

Program Description:

The subroutines are described as follows:

DBLDECIN - Waits for two decimal keys to be pressed then returns the decimal equivalent in the L register. The routine contains error trapping that will not allow a key greater than 9 or a control key to be accepted.

MULTX10 - Used by DBLDECIN to multiply the first key press by a factor of ten. This routine may come in handy in other programs.

STEPR - Moves the stepper motor one step forward or backward. The speed can be controlled by changing the label SPEED, and the direction is controlled by the value in the E register.

```
;     STEPPER MOTOR PROG

P IN        EQU   12H         ;ADRES OF PORT A
P OUT       EQU   11H         ;ADRES OF PORT B
MOS         EQU   1000H       ;MOS SERVICE
KEYIN       EQU   0BH         ;VECTOR FOR KEYIN SERVICE
LEDDEC      EQU   13H         ;VECTOR FOR LEDDEC SERVICE
SPEED       EQU   20          ;STEPR MOTOR SPEED
LEDOUT      EQU   11H



      ORG   0FF01H            ;ORIGIN OF MEM IN 8155



START:
      MVI   E,00110111B       ;THE VALUE FOR "P"
      MVI   D,1
      MVI   C,LEDOUT
      CALL  MOS

      MVI   E,11111011B       ;THE VALUE FOR "O."
      DCR   D
      CALL  MOS

      MVI   A,00110011B       ;INITIALIZE STEPPER MOTOR          ;
      STA   STEP              ;STORE IN STEP
      XRA   A                 ;CLR A REG
      STA   FINLPOS           ;CLR FINLPOS VARIABLE
      MOV   L,A               ;CLR L REG
      MOV   B,A               ;CLR B REG
      JMP   SKPCW             ;JUMP TO OUTPUT START POS TO STEPPER
MAIN:
      MOV   A,B               ;NEW POSITION BECOMES OLD POSITION
      STA   FINLPOS

      CALL  DBLDECIN          ;GET KEY BOARD VALUE

      LDA   FINLPOS
      MOV   B,A
STEPLUP:
      MVI   D,0               ;CLR D REG
      MOV   E,B               ;PLACE CURRENT POSITION ON LED DISPLAY
      MVI   C,LEDDEC
```

```
      CALL  MOS

      MOV   A,L               ;WHERE SUPPOSED TO BE
      SUB   B                 ;- WHERE AT
      JZ    MAIN              ;IF 0 EXIT LUP AND START OVER
      JC    CW                ;IF NEG GOTO CW ELSE CCW
CCW:
      INR   B                 ;INC CURENT POSITION
      XRA   A                 ;CLR A REG

      MOV   E,A               ;E = 0
      JMP   SKPCW
CW:
      DCR   B                 ;DEC CURRENT POS
      XRA   A                 ;CLR A REG
      INR   A                 ;A = 1
      MOV   E,A               ;E = 1
SKPCW:
      MVI   D,SPEED           ;SET SPEED OF STEPR
      CALL  STEPR
      JMP   STEPLUP           ;REPEAT

;*****************************************************************
;DOUBLE DECIMAL IN
;INPUT: NOTHING.
;OUTPUT: L = BINARY VALUE OF A TWO DECIMAL DIGIT INPUT FROM KEYPAD
;
;-----------------------------------------------------------------
DBLDECIN:
      MVI   B,2               ;USED AS COUNTER TO CALL KEYIN TWICE
GETPOS:
      MVI   C,KEYIN
      CALL  MOS               ;CALL KEYIN
      MOV   A,L               ;A = KEY VALUE
      CPI   10                ;IF VALUE IS > 10 ENTER AGAIN
      JNC   GETPOS
      DCR   B                 ;DEC LOOP COUNTER
      JZ    LOLBLE            ;IF ZERO THEN EXIT
      STA   HIDIG             ;IF NOT THEN STORE FIRST KEYPRESS AS
      JMP   GETPOS            ;HIGH DIGIT
LOLBLE:
      STA   LODIG             ;STORE SECOND DIGIT AS LOW DIGIT
      LDA   HIDIG             ;LOAD HIGH DIG
      MOV   B,A               ;MOV TO B
      CALL  MULTX10           ;MULTIPLY IT BY TEN
      LDA   LODIG             ;LOAD LOW DIG
      ADD   B                 ;ADD IT TO HI DIGIT
      MOV   L,A               ;STORE FINAL DEC VAL IN L
      RET
;*****************************************************************
; STEPR
; IN: D = SPEED. E = DIRECTION,1 = CW 0 = CCW
; OUT: NOTHING
;-----------------------------------------------------------------
STEPR:
      PUSH  PSW               ;SAVE A STATUS
      PUSH  B                 ;SAVE B STATUS
      MOV   A,E               ;
```

```
      RAR
      LDA   STEP          ;LOAD STEP
      JC    LEFT          ;IF E = 1 THEN GOTO LEFT
      RRC                 ;ELSE ROTATE STEP RIGHT
      JMP   SKIP          ;SKIP NEXT INSTRUCTION
LEFT:
      RLC                 ;ROTATE STEP LEFT
SKIP:
      STA   STEP          ;STORE BACK AS STEP

      IN    P OUT         ;MASK OFF 4 LSB OF OUTPUT PORT
      ANI   0F0H
      MOV   B,A
      LDA   STEP          ;LOAD STEP
      ANI   0FH           ;MASK OFF 4 MSB OF STEP
      ORA   B             ;OR WITH 4 LSB OF OUTPUT PORT
      OUT   P OUT         ;OUT STEP AS 4 LSB'S AND CURRENT STATUS OF 4
                          ;MSB'S OF OUTPUT PORT REMAIN UNCHANGED.
      PUSH  D
DELAY:
      MVI   B,0FFH        ;DELAY TO CONTROL SPEED OF STEPPER
DEL:
      DCR   B
      JNZ   DEL
      NOP
      DCR   D
      JNZ   DELAY
      POP   D
      POP   B
      POP   PSW
      RET
;*****************************************************************
;INPUT: B = VALUE TO MULT BY 10, MUST BE LESS THAN 25 DECIMAL
;-----------------------------------------------------------------
MULTX10:
      PUSH  PSW
      MOV   A,B
      RLC
      RLC
      ADD   B
      RLC
      MOV   B,A
      POP   PSW
      RET

HIDIG       DS    1
LODIG       DS    1
STEP        DS    1
FINLPOS     DS    1

      END
```