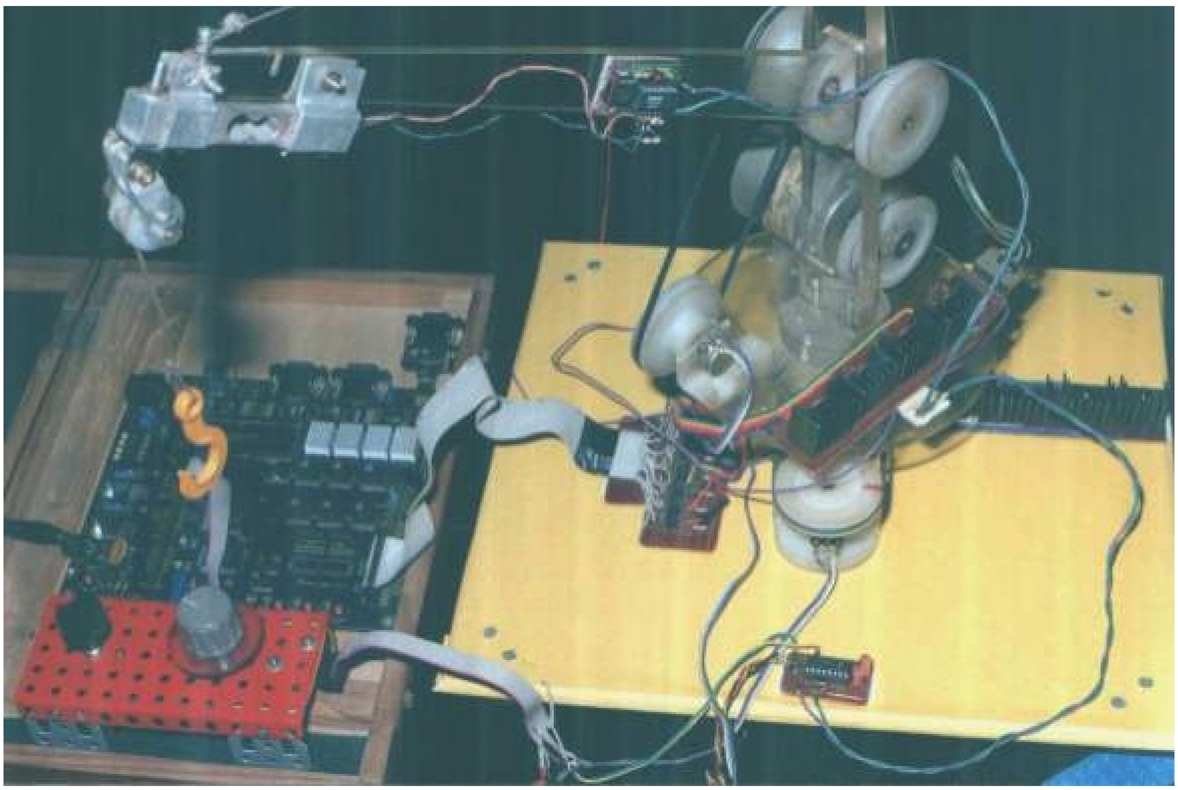


CRANE ROBOTICS CONTROLLER

[C R C]

PROJECT NO : Z4139

1. PROJECT SUMMARY (500 WORDS) :

The Crane Robotics Controller (acronymed CRC) is a Zilog, Z8F6403 microcontroller based programmable system controller for robotic control in boom raising/lowering and slew rotation in a crawler crane. The controller implements the basic philosophy involved in alt-azimuth control theory and may be implemented for a host of applications in tilt control, raising & lowering of platforms/levers, industrial robotics, etc, besides crawler cranes. The concepts are briefly, described as follows:

a) Altitude Control in boom movement with tilt or Luff angle detection, closed loop control / braking within ± 2 degrees

b) Azimuth control in rotation of base pedestal with slew angle detection & closed loop control / braking within ± 10 degrees

c) 3 digit display of Luff or tilt angle within predefined limits

d) Control panel settable alt-azimuth movements within the specified limits as programmed in the system software

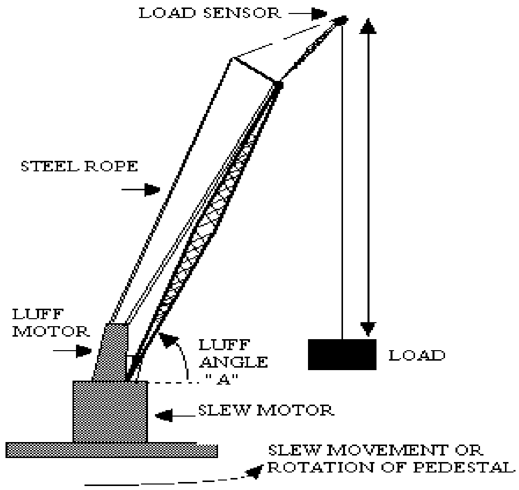
e) Overload detection and safety control during raising and lowering of loads by visual warning in case of overloading

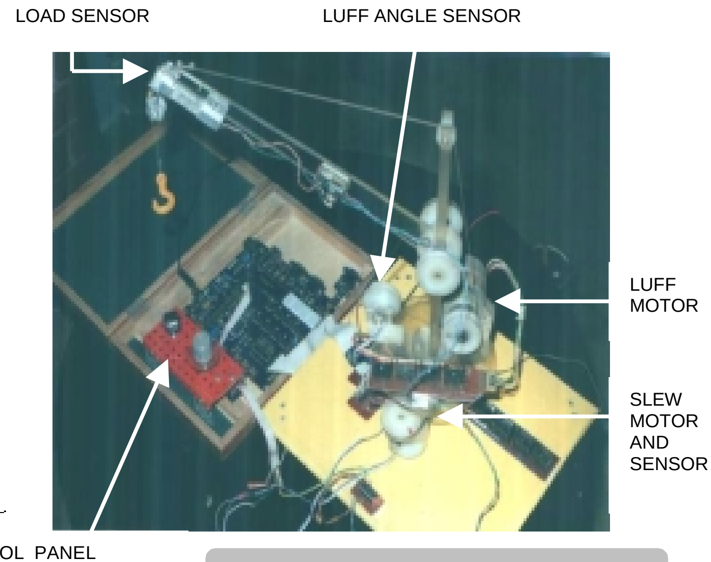
f) Load swing control during rotational movements of base pedestal

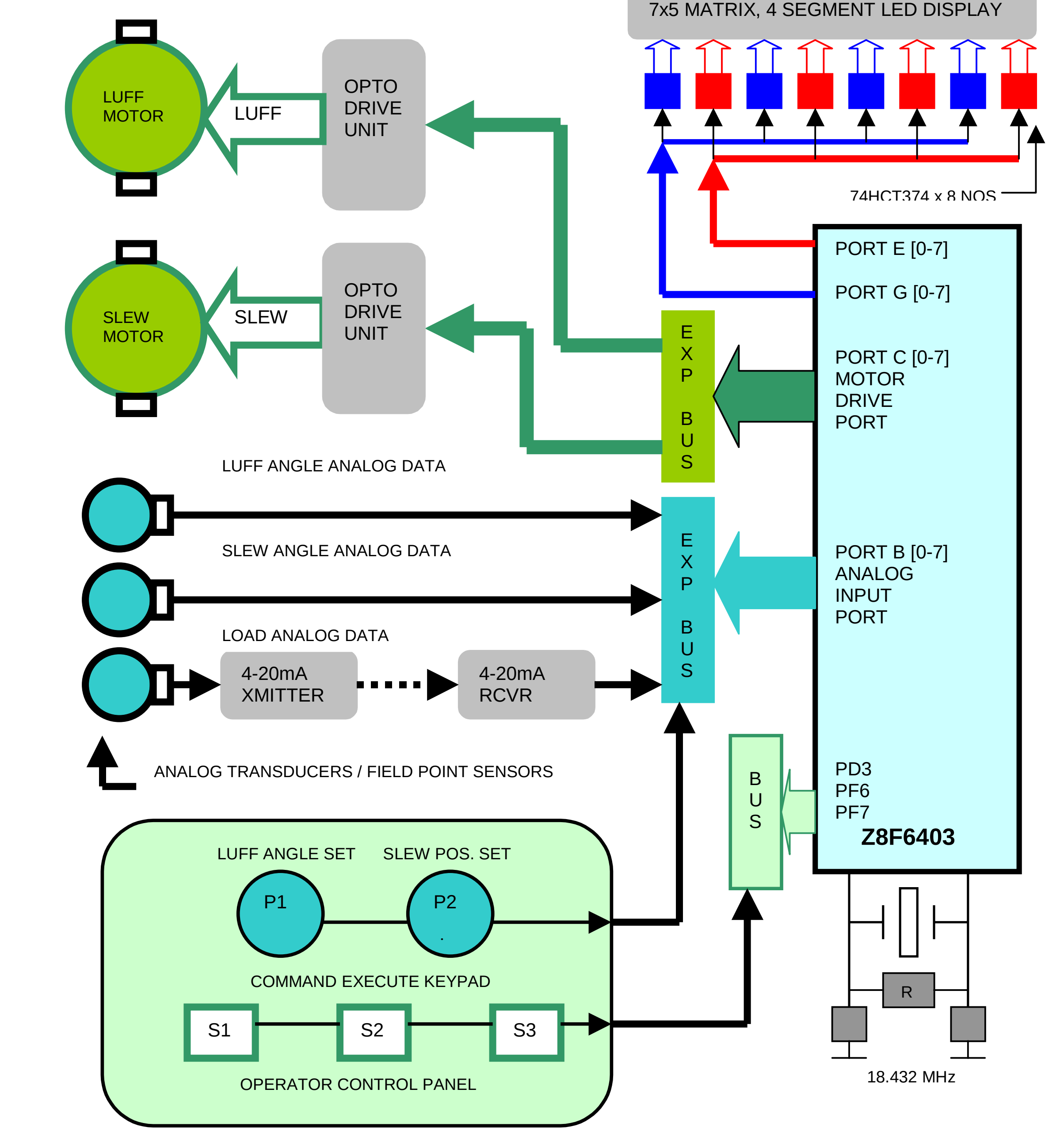
The CRC incorporates the above control features based on angle conversion tables, speed of rotation, max. & min. limits, etc. all in the onboard 64kb Flash memory and may be easily modified to suit a specific application. An important feature is the use of cost effective analog transducers to convert altitude and azimuth angle data into a linear voltage function that is AD converted for digital processing. The load detection device is a strain gauge transducer that is interfaced to a 4-20mA transmitter and the load signals transmitted over a wire loop to the controller. The 4-20mA signals are then converted into a voltage function by a 4-20mA receiver before being AD converted for further processing. Another important feature of this controller is the anti-swing control that minimises pendulum swinging of the load during slew operations and prevents accidents. The anti-swing works on the concept of the load travelling at full and half speeds while moving from one point to another either in the clockwise or anticlockwise direction. For example if the load is to move from point A to C with B as mid-point between the start and the finish, then from A to B the load moves at fullspeed and at half speed between B to C. An unique feature in this design is the calculation of the mid-point. Since the Z8F6403 MCU doesnot support a divide-by-function, a special technique has been used in the software to calculate this midpoint which makes the Z8F6403 pretty effective for divide-by-functions.

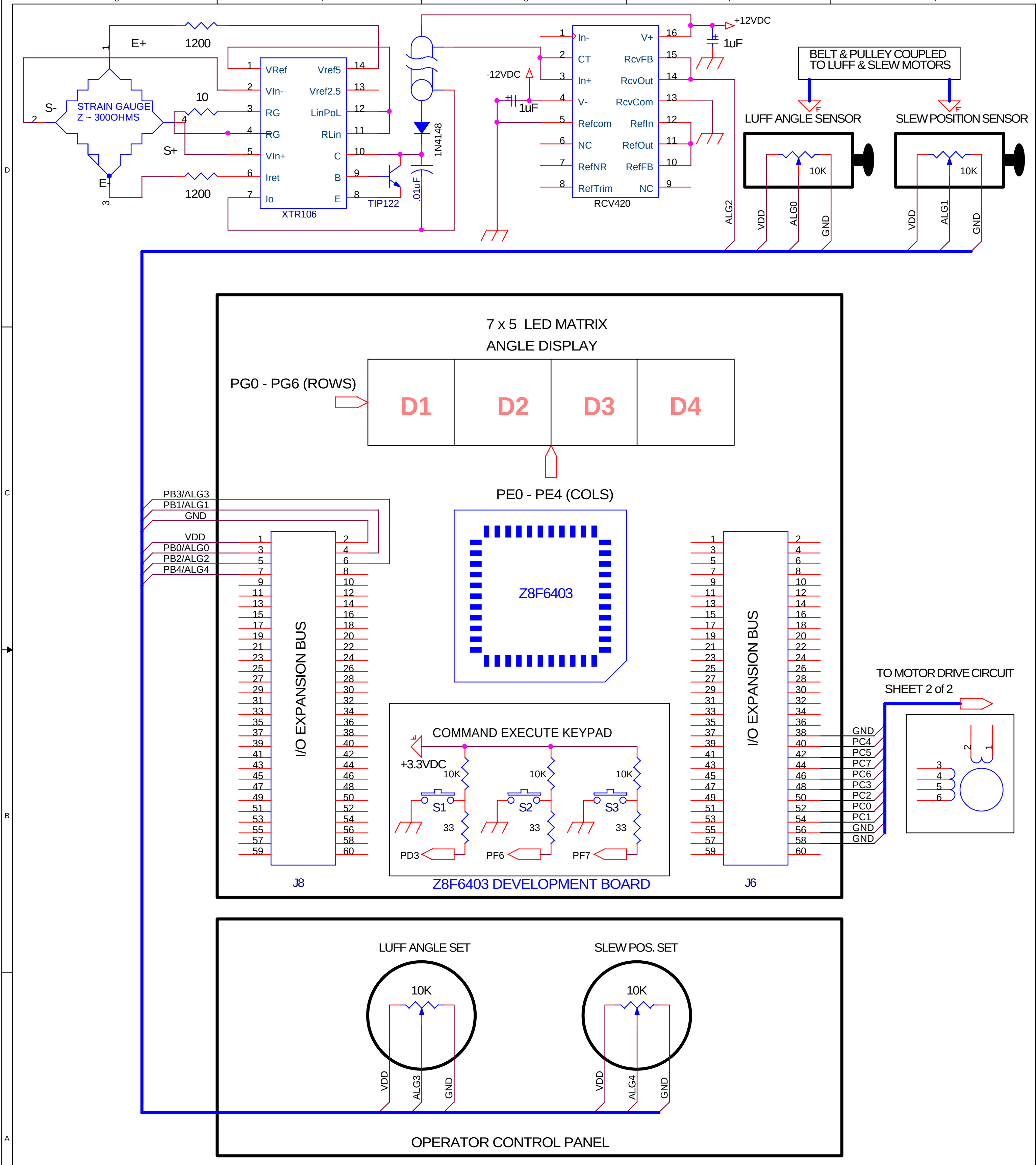
The controller is a cost effective design for control of robotic movements of crawler cranes or any type of automation system that involve alt-azimuth operations with pre-defined loads. The system offers precise movements and also incorprates the safety features of overload protection, anti-sway and is intelligent enough to initiate safety measures in case of operational mistakes. And above all the Z8Encore's high speed processing of real time data makes it ideally suited for applications in crane control.

2. SYSTEM BLOCK DIAGRAM :









DEVELOPMENT BOARD SCHEMATICS, REFER TO SCHEMATIC, Z8 ENCORE, EVALUATION BOARD, PAGES 1 TO 4, (96C0868-001 REV. C)

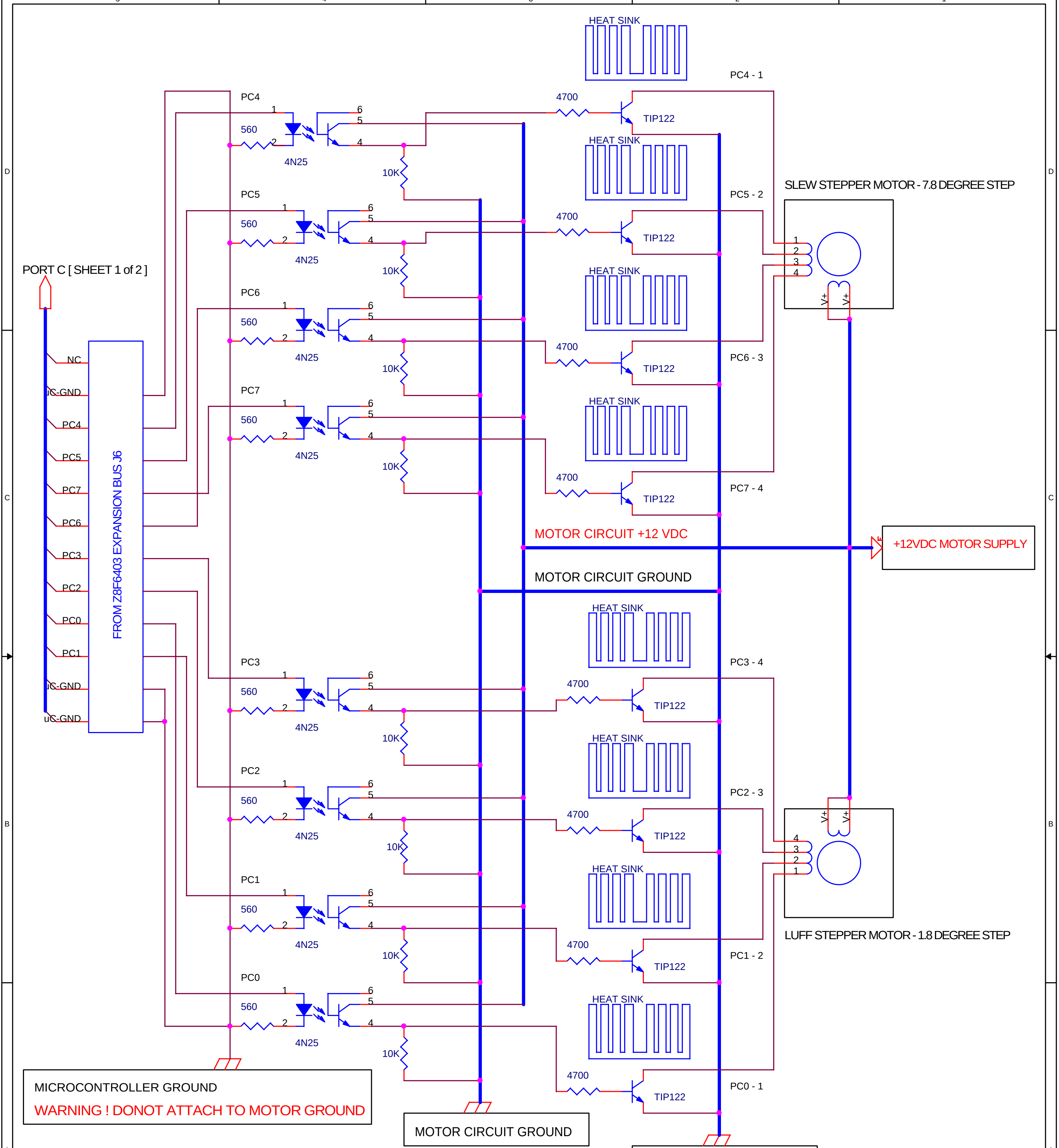
1. ALL RESISTANCES IN OHMS, 1/4 WATT
2. ALL CAPACITANCES IN FARADS
3. VARIABLE RESISTANCES ARE CARBON TYPE
4. VDD SUPPLY IS 3.3 VDC +/- 12 VDC SUPPLY FOR ANALOG/ MOTOR CKTS.
5. POWER SUPPLY TO THE DEV. BOARD IS 9 VDC

Z8 ENCORE ! INTERNATIONAL DESIGN CONTEST 2003

Title: CRANE ROBOTICS CONTROL

Size | Document Number: PROJECT NO. Z4139 | Rev: 1

Date: Saturday, October 18, 2003 | Sheet 1 of 2



1. ALL RESISTANCES IN OHMS, 1/4 WATT
2. ALL CAPACITANCES IN FARADS
3. OPTO INPUTS ARE DRIVEN BY THE MICROCONTROLLER PORT C AND ARE ISOLATED FROM THE MOTOR CIRCUIT
4. MOTOR SUPPLY 12 VDC / 3 A
5. USE HEAT SINK INSULATION KIT FOR TIP122

WARNING ! DONOT ATTACH MICROCONTROLLER GND TO MOTOR GROUND

Z8 ENCORE ! INTERNATIONAL DESIGN CONTEST 2003		
Title: CRANE ROBOTICS CONTROL		
Size	Document Number: PROJECT NO. Z4139	Rev: 1
Date: Saturday, October 18, 2003	Sheet 2 of 2	

8. DESCRIPTION OF THE CONTROLLER PROGRAM:

The preceeding sections have clearly outlined the working principles of the Crane Robotics Controller and the different functions of luff and slew operation, angle display, anti swing control and overload alerts that are available in this design. The electronic schematics clearly explain how the Z8 Encore Development board interfaced with the peripheral devices/modules serve as an intelligent controller for such operations where human supervision is not always necessary and safety features have been provided to ensure safe operation in lifting and movement of loads. The controller is basically the brain that may be used for such operations in industrial robotics, automation etc. Now it is essential to understand what goes into the brain to perform such complex tasks or else the controller program. The program is a complex combination of different sub-routine calls with polled interrupts added to the main routine and is as follows :

a) Main Routine (main.asm):

After a Power-ON Reset, the reset vector originates at address 0002Hex and starts the execution of instructions from label RESET. Interrupts are disabled as this program does not use vectored interrupts and only polled interrupts are used. The register pointer and the stack pointer are set and the the program calls the general purpose I/O initialisation sub-routine, **init.asm** to configure the Ports E & G for the display, Port C for motor drive and Port F&D for the command execute keypad.

```
( main.asm )
; PROGRAM START. . .
;*************************************************************
      org  %100
RESET:
      DI
      SRP    #%E0
      LDX    xsph,#high stack_top
      LDX    xspl,#low stack_top
      CALL  INIT_GPIO           ; SUB-ROUTINE
      CALL  INIT_DMA_ADC        ; SUB-ROUTINE
```

Next is the sub-routine call for initialising the DMA_ADC. The DMA_ADC is configured for ADC value transfers from Port B, **analog inputs channels ALG0 - ALG4 to the memory locations from DE0 to DE9,** each channel occupying two consecutive memory locations for storing the 10 bit ADC value. In this design the degree of accuracy in load and angle data is achieved only with the uppper 8 bits of the ADC value and the lower 2 bits are discarded. For a 10 bit ADC, the resolution is 2 mV whereas if the upper eight bits are taken, the resolution comes to around 8 mV which is sufficient for the level of accuracy required in this controller. For example if the total angular displacement of the sensor potentiometer is 270 degrees then for a 10 bit ADC we have a resolution of 0.26 degree whereas an 8 bit ADC yields a resolution of 1.1 degree. In a mobile/fixed crane because of the inertia a braking accuracy of .26 degree is but an utopia and even a 1 degree accuracy is not possible. However, for the test bench model used in this project, the stepper incerment is in steps of 1.8 degrees and hence the resolution of 1.1 degree with an 8 bit ADC is well within the limit.

```
INIT_DMA_ADC

     LDX  DAADDR,#DEH     ; dump adc values from DE0
     LDX  DACTL, #84H     ; ALG0 TO ALG4 enabled
     RET
```

After RESET and initialisation the Mainloop starts at the START label as shown in the snippet. The mainloop is termed a loop because after executing a sub-routine the PC jumps back to START to perform the standard tasks. The first task is to load the register R8 with the AD converted value from the location DE4h which corresponds to the upper 8 bits of the Analog to digital conversion (ADC) value from Port B analog input ALG4 stored in that location.

```
;*************************************************************
;  MAINLOOP PROGRAM
;*************************************************************
START:
      LDX    R8,  DE4h
      CP     R8,  #23h
      JP     UGT,  OVERLOAD
      LDX    IRQ0, #00h
      LD     R9,  #00h
      LDX    R5,  FE6h
      BTJZ   7,R5, SLEW_CALL
      LDX    R5,  FDEh
      BTJNZ  3,R5, NORMAL
```