1982

Micro-computers, slotcars and education

P. J. McKerrow
University of Wollongong, phillip@uow.edu.au

MICRO-COMPUTERS, SLOTCARS AND EDUCATION

Phillip John McKerrow

Department of Computing Science,
The University of Wollongong,
Post Office Box 1144,
Wollongong, N.S.W. 2500
Australia.

ABSTRACT

Working models interfaced to micro-computers provide an inexpensive way of simulating real-world situations. Students working with these systems gain a feel for the scope of micro-computer applications in addition to learning micro-processor fundamentals.

A micro-computer controlled slot car system has been built for use in the laboratory section of a micro-computer course. This system allows the students to experiment with a realistic micro-computer application. A stimulating environment, where learning is enjoyable, has been created.

Keywords: Micro-processor, micro-computer, slotcar, education, real-time, programming, hardware, software, interface.

1. Introduction

Micro-computer courses are growing in popularity as a consequence of the rapidly increasing application of micro-processors to virtually every area of life. In the real world of micro-computing the distinctions made on large systems between hardware, software and applications are giving way to integrated designs. Increasingly micro-computer practitioners have to be competent in all areas, requiring courses which give a full orbed coverage of the field.

Universities tend to produce computing scientists who are expert programmers but who have little understanding of real-

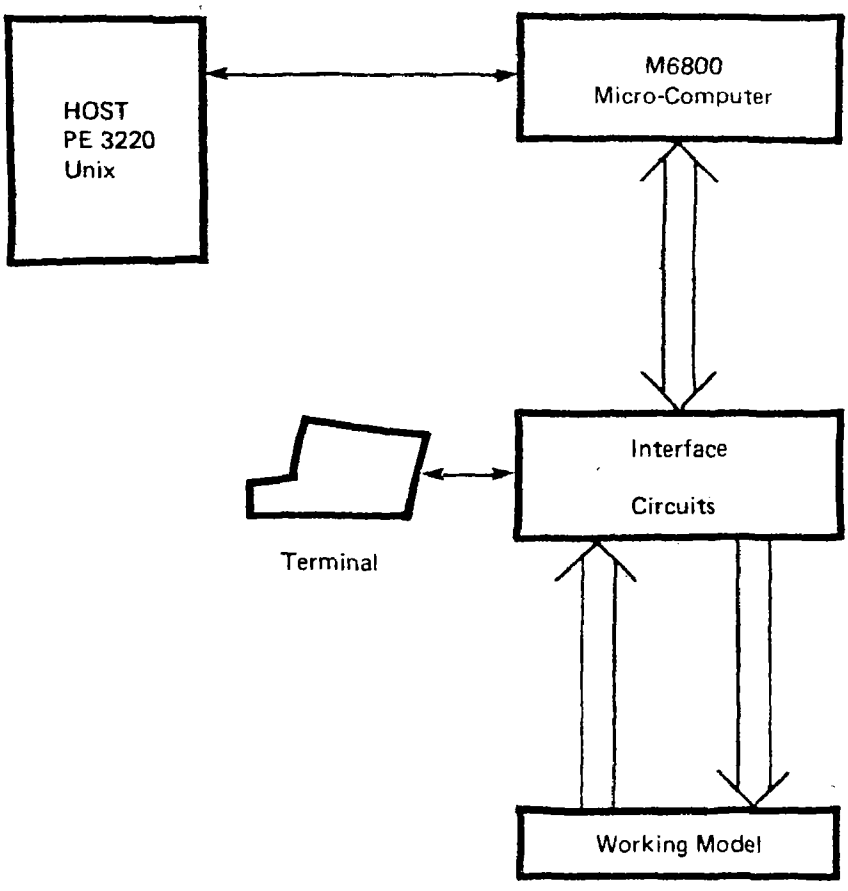

FIGURE I TYPICAL MICRO-COMPUTER LABORATORY SYSTEM

world applications. This is particularly true in the field of process control where students not only lack experience in control theory but also in fundamental physics.

Working models provide an inexpensive way of simulating the real-world. They can be built from readily available hobby equipment and interfaced to micro-computers using standard peripheral chips and transducers. Hands-on experience obtained by working in a simulated real-world environment enable students to gain a feel for the scope of computer control. They learn both the limitations and possibilities of micro-computing. Interest in tutorials is increased because the results are observable and often fun. Getting a computer to successfully manipulate the physical world gives students a real sense of achievement and failures lead to understanding.

Laboratory experiments which include computer controled models, can be used to teach microcomputer hardware and software, interface circuitry, process control and real-time computing. Systems developed, as part of the micro-computer laboratory include:

(i) an electronic dice, using a parallel interface adapter, a push button and a seven segment display,

(ii) a transparent link, using two asynchronous communications interface adapters; one connected to a terminal and the other to a host computer,

(iii) a square wave music system, using programmable timers to drive the speakers,

(iv) a Hewlett Packard bar code reader for universal product code reading,

(v) a terapin turtle robot, and

(vi) a slot car control system.

2. Microcomputer System

The most sophisticated working model is the slot car system where the microcomputer program has to both monitor the position of the cars and control their speed. The hardware for all these systems has been designed and constructed by departmental staff. The students have to write programs to make the hardware work requiring thorough knowledge of all areas of the system.

Each laboratory system (figure 1) consists of a microcomputer linked to a host computer, interface circuits and a working model. Programs are edited and cross compiled on the host computer and then down line loaded to the target microcomputer. Students can do most of their development work on the host, using tools they are familiar with. Access to the microcomputer is

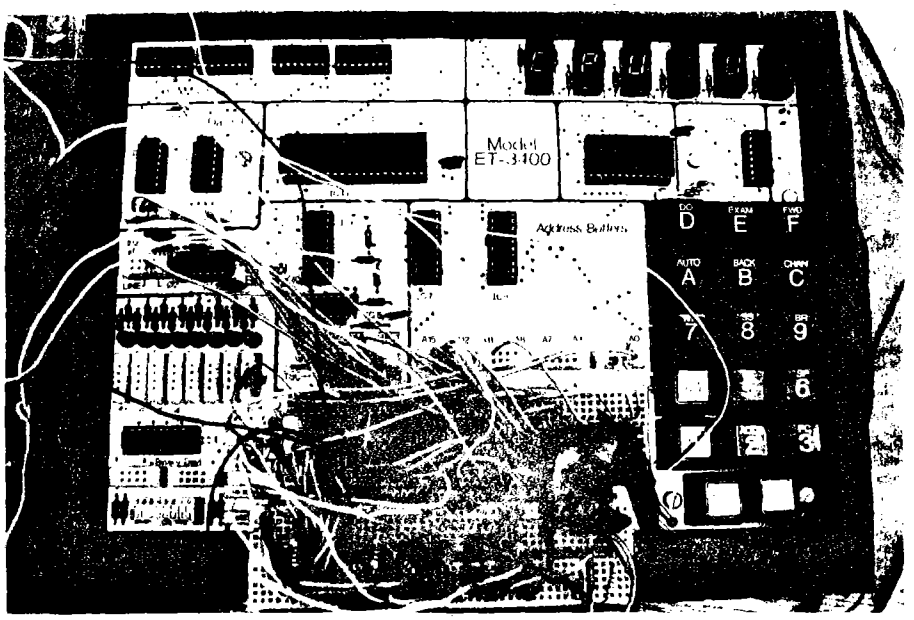

Figure 2. Original Interface Circuits

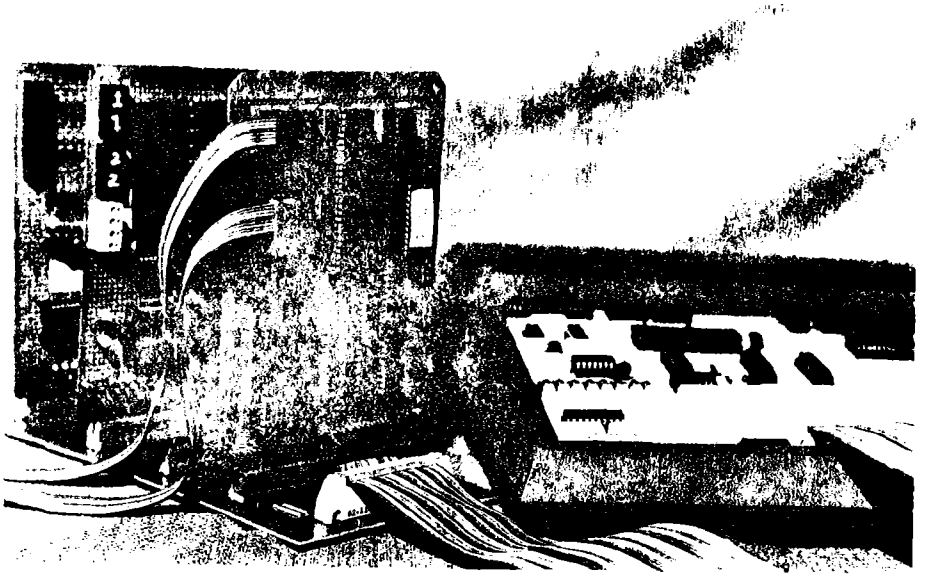

Figure 3. Micro-computer expansion system including bus expansion, mother board and interface cards.

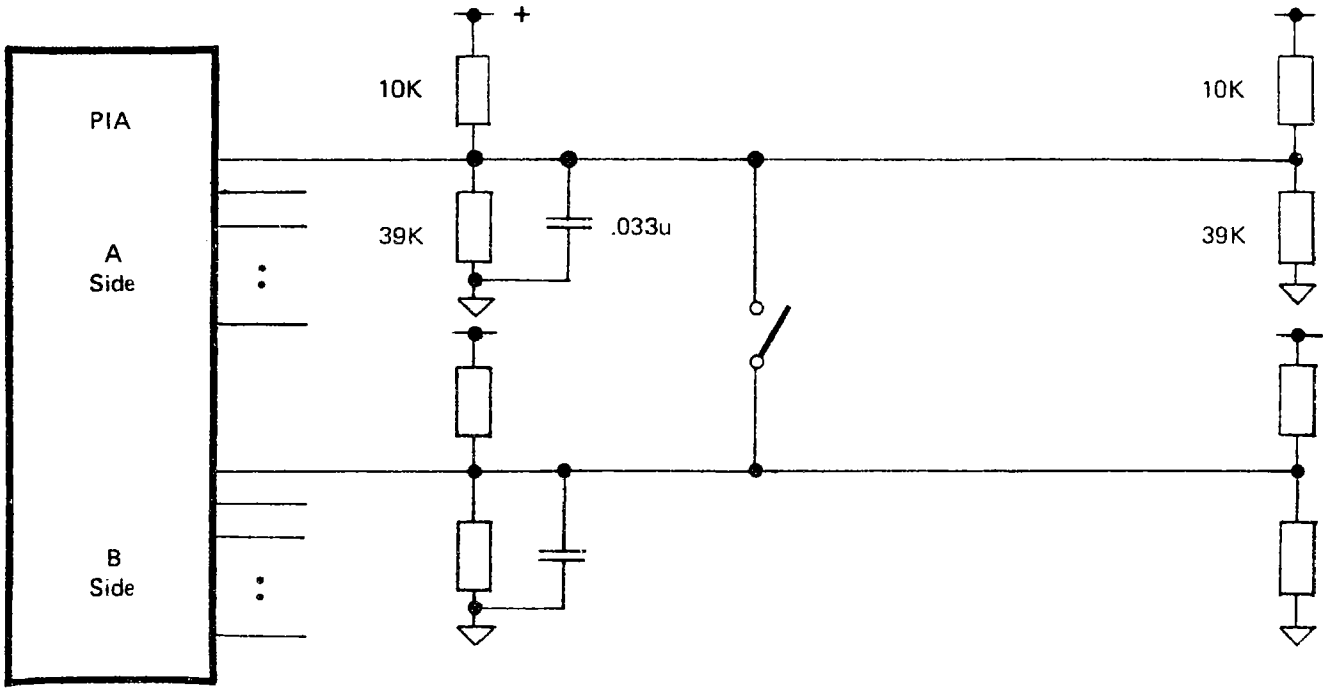

FIGURE 4 KEY SWITCH MATRIX (8 X 8)

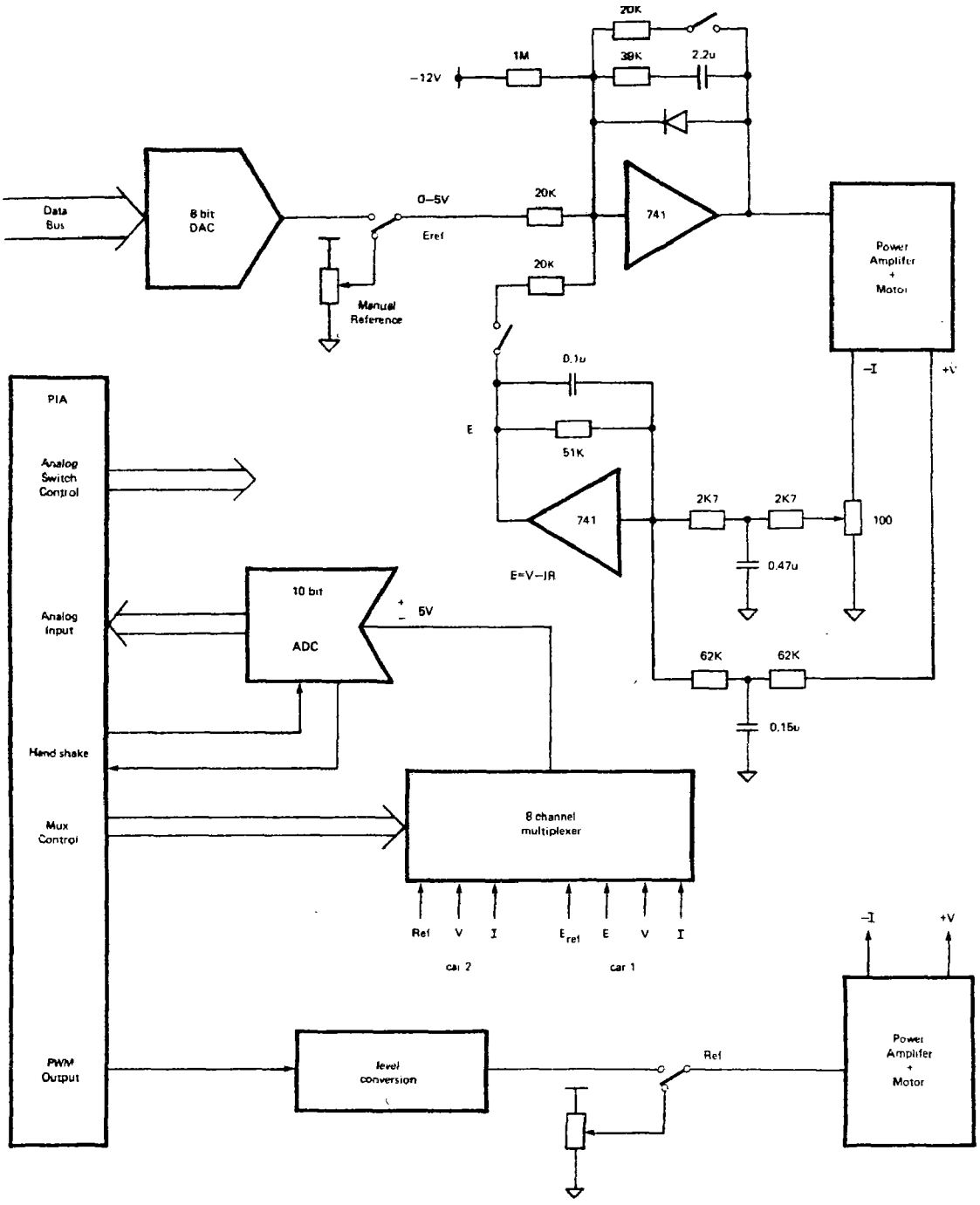

FIGURE 5 CAR SPEED CONTROLLERS – ANALOG AND PULSE WIDTH MODULATED

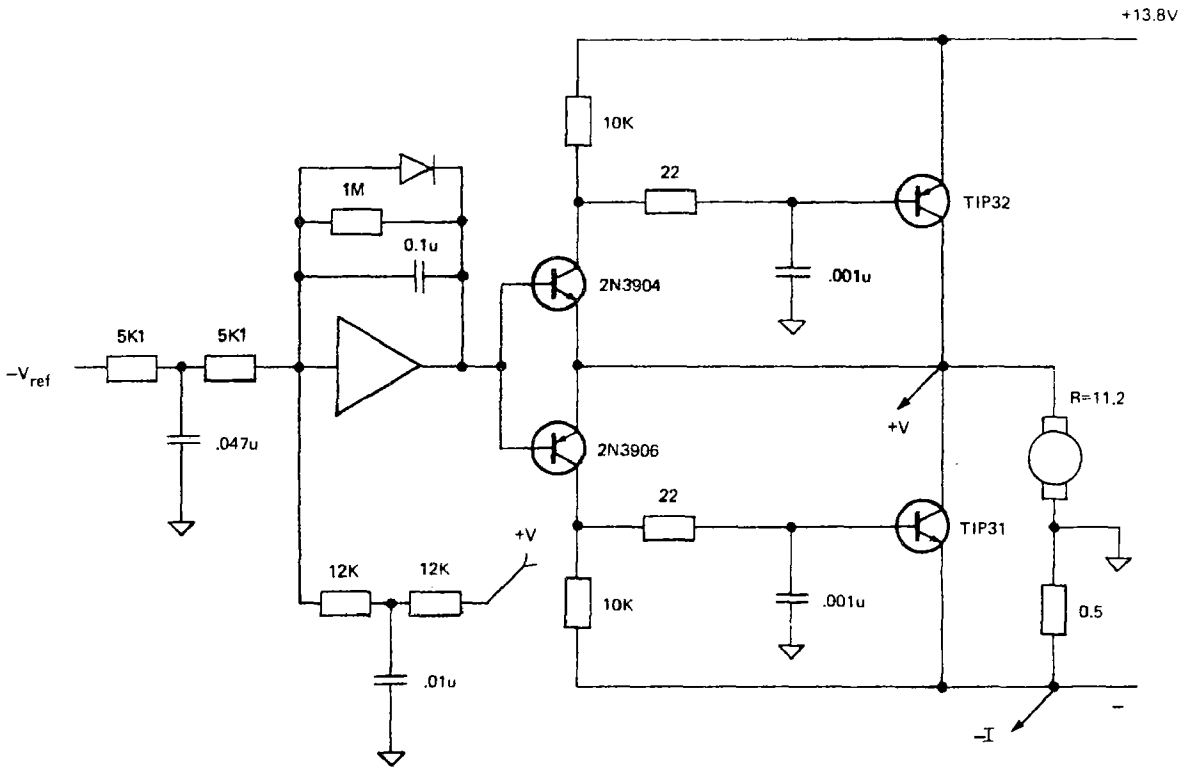

FIGURE 6 CLASS B POWER AMPLIFIER.

only needed for program testing and debugging, reducing the number of microcomputer systems needed.

A Motorola M6800 microprocessor system is used. Originally interface circuits were hay wired on the prototype section of the Heathkit trainer (figure 2). Recently a trainer expansion system (figure 3), which includes bus expansion, mother board, and wire wrapped memory and interface cards, has been built.

3. Slotcar Control

Aurora AFX G+Plus cars were chosen because their motors have a very strong magnet in them used both for propulsion and holding the car on the track. The magnet will operate a reed switch up to 12 milli-metres under the track. Reed switches, mounted at appropriate positions under the track, are wired back to a switch matrix (figure 4). Key closure is detected under software control. The position monitoring program can either scan the matrix rows or can benefit from the dynamic characteristics of the programmable interface adapter by using line reversal techniques [1] to read the matrix more quickly. A software algorithm similar to n-key rollover is required to compensate for ghosting caused by multiple key closure. The program must wait for one milli-second between keyboard reads to allow the debounce filters to settle.

Two car speed control circuits are used. One is an analog controller using a digital to analog converter to drive a power amplifier (figure 5). Initially the car was on open loop voltage control but frictional losses vary considerably as the car warms up. Now current feedback is used to calculate an electro motive force signal (which is proportional to rotational speed of the motor) for closed loop control. The loop can be closed externally using preset timing components or closed in the software using direct digital control depending upon the purpose of the tutorial.

In the second controller, a software generated pulse-width-modulated signal drives the power amplifier. Closed loop control has to be effected in software. Both controllers use a class B power amplifier (figure 6) which allows regenerative braking of the car. Initially an open-collector power-stage was used but when decelerating the energy in the car was only dissipated in frictional losses and the car took excessive time to slow down. When using the normal hand controller, energy is dissipated in the rheostat.

4. Design Problems

When we built the first track layout (figure 7) we did not realise how real-world the model is. The car motors generate an incredible amount of electrical noise. Most of the noise problems were overcome by solidly earthing everything. All external wiring, particularly to the read switches, had to be heavily filtered and terminated to reduce both noise from the motors and

Figure 7. Initial track layout

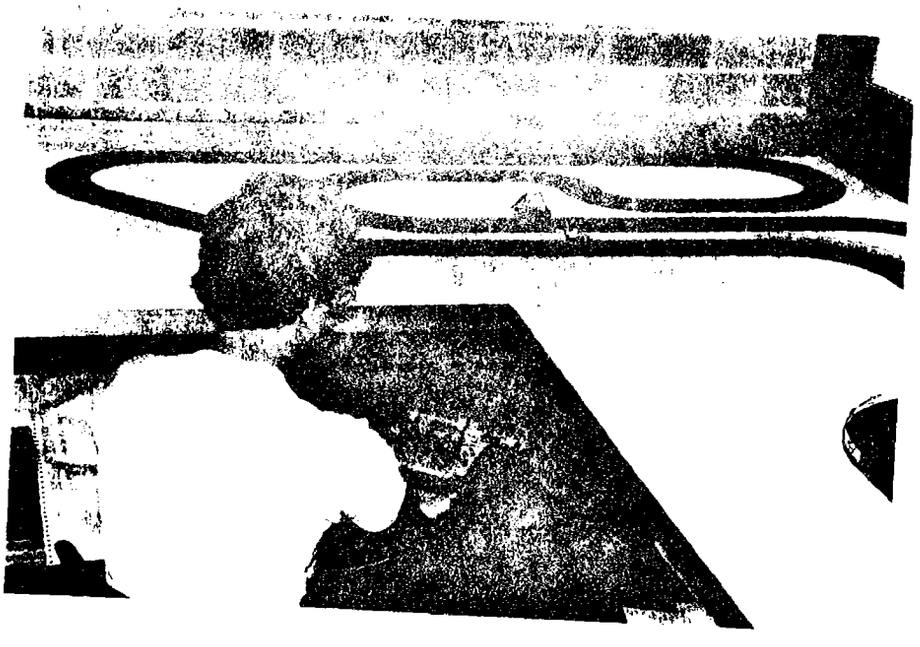

Figure 8. Final track layout

```
Set analog switches for car 1 to select computer speed reference
and external closed loop control.
Set 8 channel multiplexer to select car 1 emf.

Repeat
      Request analog to digital conversion.
      Wait for conversion to complete.
      Read car 1 emf.
      Display car position,  speed reference and emf.

      While matrix switch not closed do   {scan matrix}
           Select next matrix output.
           Wait 1 millisecond.   {allow debounce filter to settle}
           Read matrix input.

     Wait 2 milliseconds   {switch closed}
     Read matrix input again   {software debounce}

     If switch closed then
           Case switch meaning of
           1.   start car        {output initial speed reference}
           2.   stop car         {output zero speed reference}
           3.   car detected     {calculate  index  into speed
                                  table, obtain speed for next
                                  track  section   and  output
                                  speed reference}
           4.   invalid          {e.g. ghosting - ignore switch}

until stop
```

Figure 9. Algorithm of simple speed control program for one car.

crosstalk from other read switches closing. The power supply, included in the set, was unregulated and had considerable droop, consequently when one car accelerated the voltage drooped affecting the speed of the other. One car decelerating could cause the other car to leave the track if that car was rounding a corner. This has been overcome by installing separate power supplies for each track.

Both car and track are of high quality and easy to service. The car needs regular oiling and internal cleaning. As the car tyres wear down the ratio of car linear velocity to motor rotational velocity drops but as the car is closer to the track the magnetic attraction increases allowing higher speeds until finally the car scrapes on the track.

The solid silver-plated pickups are very effective but three track conditions cause problems:

(i) Resistive track to track connections cause a decrease in the voltage available to the motor.

(ii) Track electrical rails should protrude 0.5 to 1 millimetre above the track surface. If they are below the surface electrical continuity is lost. Normally the condition can be rectified by pushing the rail, from the back of the track, until it is at the correct height.

(iii) The track rails oxidise rapidly producing an insulating layer of oxide between the rails and pickups. This can be overcome by regular track cleaning with an ink eraser or by driving a car around the track, with a hand controller, for a few laps.

A range of track sections are available allowing complex track layouts. We have gone away from our initial layout which included a fly-over to a flat layout (figure 8) because some configurations create problems for both track maintenance and computer control. These include; curves greater than 180 degrees, flexi-track, banked curves and criss-cross sections.

5. Control Software

Currently the system is used in a microprocessor course [2]. Students are required to write an assembler program to detect the car position and set its speed.

The simplest program (figure 9) scans the switch matrix for a read switch closure. When a car is detected the switch position is encoded and used as an index into a look up table to obtain the speed value for the next section of track. Speed values are set into the table manually.

More sophisticated software includes controlling the speed

of one car to be the same as a human driven car on the other track, measuring car speed and setting up the look up table, speed iterations between sections of track, direct digital control of the motors etc. A wide range of interesting student assignments is possible.

6. Conclusion

Working models interfaced to micro-computers provide an inexpensive way of simulating the real world. Students working with these systems gain a feel for the scope of computer control as well as learning the fundamentals of micro-computer hardware, software and interfacing techniques. A micro-computer controlled slot car system enables the student to experiment with analog and digital input/output, optimisation techniques, real-time computing and applied control theory. These factors combine together to provide a stimulating environment where learning is enjoyable.

7. Bibliography

1 R. Zaks and A. Lesea, Microprocessor Interfacing Techniques, Sybex, 1979.

2 L. A. Leventhal, Introduction to Microprocessors: Software, Hardware, Programming, Prentice-Hall, 1978.