technical note

Int J Appl Radiat Isot Vol 36, No 11, pp 908–910, 1985
© Pergamon Press Ltd 1985 Printed in Great Britain
0020-708X/85 $3 00 + 0 00

Microcomputer Control of Radiochemical Processes

STUART C SWEET and
HENRY C GRIFFIN*
Department of Chemistry, University of Michigan
Ann Arbor, MI 48109, U S A

(*Received* 11 *October* 1984,
in revised form 15 *March* 1985)

A system for control of radiochemical processes has been designed and constructed Control passes through a single 8-bit port (6522 VIA) to up to 64 on–off sensors and 64 on–off switches The outputs are latched, they are switched one at a time

Introduction

Transforming the radiochemical manipulations developed in the laboratory to "production" operations is strongly dependent on automation This dependence derives from the needs for reproducibility in the radiopharmaceutical products, low radiation exposure for production workers, and (for short-lived nuclides) optimization in timing in the overall process There are secondary advantages, such as improved documentation of individual batches The primary disadvantage of automated radiochemical processing is that many standard laboratory operations—particularly transfers of solids, such as addition of powders and collection of precipitates—do not admit easy automation

Systems based on microcomputers are programmable (i e conveniently changeable in the sequence and timing of unit operations) A fundamental limit of a microcomputer is the amount of information it can process at one time (in "parallel") Although 16-bit computers are entering the price range acceptable for controllers, more commonly the low cost computers are 8-bit devices That is, the information handled in a single step might correspond to the states (open or closed) of eight switches This is an example of "input"—information coming from the laboratory to the computer Similarly, 8 bits of "output" might be commands to eight valves to open or close In order to overcome this limitation we have constructed an interface which uses multiplexing logic to control up to 64 input and 64 output bits

Dedicated, microprocessor-based controllers are available commercially Some of these are quite inexpensive, but our interest in documentation as well as control ruled out these devices In designing the overall system we have benefited from reports of remotely-controlled radiochemical procedures[1-5] These are "manual" systems as opposed to computer based, but Davis *et al*[2] used a complex switch with parallel control of 15 components

The initial designs for radiochemical procedures to be used at the University of Michigan Cyclotron/PET Facility clearly required at least 15 switches and 10 test points The test points correspond to a variety of sensors or transducers, and in some instances finer distinctions than yes/no or open/closed would be desirable In addition, the valves and other typical devices receiving commands from the computer are not directly responsive to the signals (TTL levels) the computer can produce, level translations (drivers) are needed Therefore, a complete system involves more than just transmission of information However, even simple chemical processes may require some multiplexing in order to interface to more than 8 control or monitoring points

Interface Design

Logical design

Details on the 6522 interface "chip" and machine language programming of the 6502 microprocessor have been described by Leventhal[6] The principle of the design is to permit many input or output connections one at a time rather than a few simultaneously The 8 I/O lines are logically divided into six address lines (64 addresses), one bank select line (to select the input bank or the output bank), and one data line The six address lines are subdivided into three lines to select 1 of 8 groups (chip select) and three lines to select 1 of 8 lines through a given chip (line select) In practice we have used a somewhat less direct approach for bank selection and data (We wanted to reserve one of the I/O lines for future "global" functions such as an emergency interrupt to meet conditions not anticipated in an automation program) The design of the interface is given in Fig 1 The lower part of the Figure corresponds to the output bank and the upper part to the input bank Line select and chip select correspond to PA0-2 and PA3-5, respectively, from the computer For output (but not input) CB2 is the bank select line and PA7 is the data line For input PA7 is the bank select line and CA1 is the data line into the computer

Setting output lines

Setting an output line is somewhat simpler than reading an input The computer program which accomplishes output is given in Fig 2 (Refer to Fig 1 for definitions of the various lines) The operations assume that prior steps have placed the output bit address (line 38 out of the 64 possible output lines) and data (high in this illustration) in location 1000 (hexadecimal address $03e8) The computer is configured with the 6522 VIA occupying computer addresses 59456 ($e840) through 59471 ($e84f) Bit 4 of a control register in location 59468 controls CB2 The first two lines of machine code define constants which will be used to control CB2 The third line insures that CB2 is low The fourth line gets the information concerning which output line is to be set and which state is to be set Line five sets the address and data, line six strobes the latch so selected, line seven disables the setting mode (leaving the output line "latched" in the desired state), and the last line returns computer operation to other processes

Reading input

Reading an input requires extra steps to set, read and reset the interrupt register In short, an input is read by determining whether a transition is seen on CA1 when the addressed line is connected into that line

Use of PA7 for both input and output operations does not cause conflicts The fact that data may be presented to pin 3 of the latches is not important except when the latches

* Author to whom correspondence should be addressed

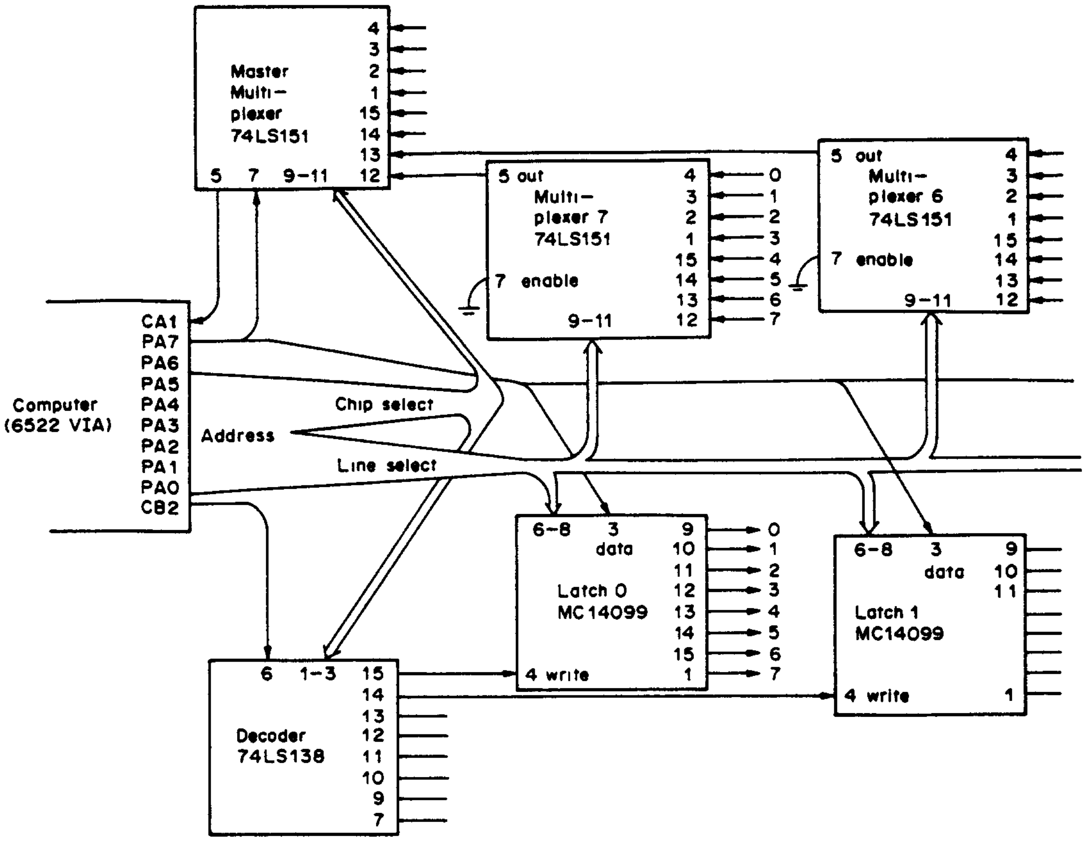

Fig 1 Diagram of the interface

are enabled (by CB2) Conversely, although the data used for output might enable the input bank, any transition on CA1 is ignored

We have found that I/O operations can be carried out roughly 1000 times faster when controlled by a machine language subroutine as opposed to comparable BASIC statements For this reason we have chosen to handle all interface control with machine language subroutines

Output driver

The devices which are controlled in our system require a 12 V power source This power is switched by the circuit given in Fig 3 Each device has a control as given in the right side of the Figure The left side permits switching to manual control, which has been found quite useful in setting up chemical procedures When the relay is in the position indicated in Fig 3, the mechanical switch in the device

Setting an OUTPUT Line

MACHINE LANGUAGE

```
af cf      LDX  #207
a0 ef      LDY  #239
8e 4c e8   STX  $e84c    Set CB2 low
ad e8 03   LDA  $03e8    Get address+data*128
8d 4f e8   STA  $e84f    Put address on PA0-7
8c 4c e8   STY  $e84c    Set CB2 high
8e 4c e8   STX  $e84c      ,,   ,,  low
60         RTS           Return from subroutine
```

BASIC

```
POKE 59468,207
POKE 59471,PEEK(1000)
POKE 59468,239
POKE 59468,207
RETURN
```

Fig 2 Commands for setting outputs

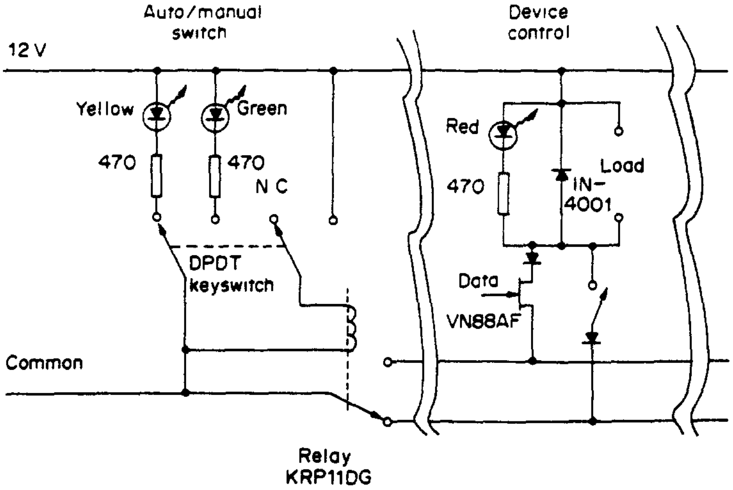

Fig 3 Output drivers

control section controls power delivered to the device When the relay is in the other position, each device is controlled by "data" from the computer to the gate of a VN88AF transistor The indicator light-emitting-diodes facilitate interconversion between manual and auto control, by flipping the auto/manual switch one can confirm that a computer setting is the same as the manual setting

The input circuits require no drivers In effect each input device is a SPST switch connecting a particular multiplexer line to ground CA1 senses when an addressed input is at ground potential

Discussion

We have used the "8 to N" interface for over 2 years and with several different radiochemical procedures The controller is a Commodore 8032 computer, which has an internal clock for timing the outputs and logging the inputs This information is stored on "floppy" disks for documentation of each run The system is quite powerful in spite of its relatively low cost Essentially the same interface and control programs could be used with even less expensive VIC 20[7] and Commodore 64[8] computers More complex parallel I/O can be easily added to this system by dedicating some of the 64 output bits to controlling I/O buffer chips

Acknowledgements—We thank D Jewett and R Ehrenkaufer for providing specifications for the radiochemical procedures

References

1 Laufer P and Kloster G *Int J Appl Radiat Isot* **33**, 775 (1982)

2 Davis J, Yano Y, Cahoon J and Budinger T *Int J Appl Radiat Isot* **33**, 363 (1982)

3 Berger G, Maziere M, Knipper R, Prenant C and Comar D *Int J Appl Radiat Isot* **30**, 393 (1979)

4 Van Haver D, De Clercq P, Vandewalle T and Vandecasteele C *Int J Appl Radiat Isot* **33**, 751 (1982)

5 Ruth T J, Lambrecht R M, Wolf A P and Thakur M L *Int J. Appl Radiat Isot* **31**, 51 (1980)

6 Leventhal L A *6502 Assembly Language Programming* (Osborne/McGraw-Hill, Berkeley, 1979)

7 Finkel A, Harris N, Higgenbottom P and Tomczyk M *VIC 20 Programmer's Reference Guide* (H. Sams, Indianapolis, 1982)

8 *Commodore 64 Programmer's Reference Guide* (Commodore Business Machines, 1982)