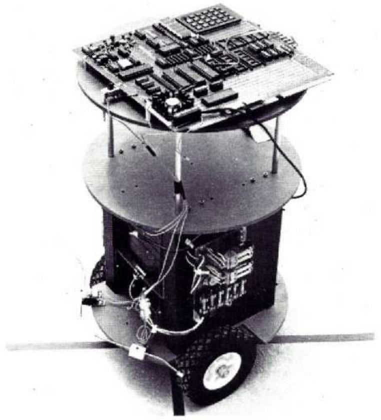
Don McAllister
4709 Rockbluff Dr.
Rolling Hills Estates, CA 90274

SUPERKIM Meets ET-2



This article presents some of my experiences in interfacing and programming a SUPER-KIM single board computer (SBC) for the control of a Lour Control ET-2 robot shell (Figure 1). The ET-2 (*E*xperimental *T*ransmobile with *2* drive motors) consists of a three level frame powered by two separately driven wheels and balanced by a free caster. The lower level contains the drive motors and gearbox, a 32 amp-hour 12V motocycle battery, and two driver electronics boards. The upper levels are available for the installation of user equipment. In this case, the SBC is mounted on top.

The ET-2 may be operated under computer control using only four TTL command lines. Each motor has two control bits, one to turn it on and another to set its direction (by a reversing relay). The driver boards provide the amplification necessary to convert from TTL logic levels to the 12 volt power for the motors and relays. Control of motor speed is obtained by varying the duty cycle (the percentage of time the bit is on) of a low frequency (10-20Hz) square wave signal applied to the motor's drive bit. The inertia of the motor and robot effectively average the signal to a proportionally lower DC level at the motor.

The drive motors are Ford permanent magnet windshield-wiper motors, which, besides having built-in gear reduction that produces a good deal of torque, are also less expensive than PM motors with comparable performance and are readily obtainable. Each motor can be independently driven in the forward or reverse direction. Lour states that to turn the shell, the preferred method is to drive one motor forward and the other in reverse so that the robot spins on its vertical axis. Turns with only one motor driving are not recommended, due to the increased loading of the motor. Reversing a motor while it is in operation can put a tremendous strain on the motors and drive system. Thus, both motors should be programmed to stop briefly between commands.

The SUPERKIM, by Microproducts, Inc. is a complete, powerful microcomputer control system based on the 6502 microprocessor, contained on a single 11.5 x 11.5 inch PC board. The board is fully socketed for easy servicing and expansion to 4Kbyte RAM and 16K EPROM on board. It comes with 1K RAM, and the address space is fully decoded so that with additional boards up to 64K of memory or I/O may be used. For this purpose, the CPU bus lines are brought out on wire-wrap pins that may also be used with standard in-line ribbon cable connectors to expand the bus.

The SUPERKIM has eight priority interrupts which are individually vectored and resettable under software control—a feature useful for real-time robot control systems. Four SYNERTEK 6522 Versatile Interface Adapter (VIA) sockets are provided on the board; one 6522 comes with the board. This IC is indeed a very flexible I/O device, containing two bidirectional 8-bit parallel ports with handshaking (with each bit separately programmable for input or output), an 8-bit, bidirectional serial to parallel shift register, and two 16-bit programmable counter/timers. The board comes with a 6530 interface chip as well.

The ports on the 6522's could also be used for implementing analog to digital converters (ADC's). A full complement of 6522's would permit up to eight 8-bit ADC's for interfacing to robot sensors, etc.

Interfacing the SUPERKIM to the ET-2

The SUPERKIM is mounted to the topmost PVC platform on the ET-2 with machine screws and .75" spacers. 12V from the battery is supplied to the SUPERKIM's on-board 5V regulator through a SPDT switch.

Figure 2 shows the location of the pins on the 6522 that are used as output ports to the ET-2. The four control lines of the ET-2, D1, D2, E1, E2, are connected to the control bits in the SUPERKIM's J5 VIA parallel output port as shown in Table 1. There are, of course, many alternate possibilities for configuring the interface. For convenience, the motor drive signals were assigned to bits 0 to 1 of port A (address 1302H) and the reversing relays to the corresponding bits of port B (address 1303H). Note that since the drivers on the ET-2 invert the logical sense of their inputs, a logical 0 (low) on an output will turn the corresponding motor or relay ON, and a logical 1 (high) will turn it OFF. Thus, writing to addressed 1302 and 1303 controls the motors and relays directly, with sixteen possible control states.

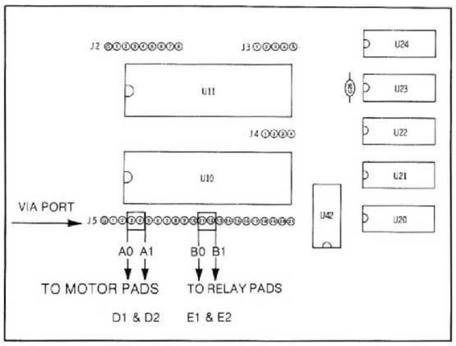


Figure 2. SuperKim/ET-2 robot control interface connections.

Due to the action of the power on reset, the I/O ports of the 6522 are initialized to be output ports, and zeroed. Therefore, as soon as the SUPERKIM is turned on, the ET-2 will lurch forward if the motor drivers are connected to the interface. To eliminate this problem, a 2-pole switch is used between the 6522 outputs and the motor drive inputs, which should be open when the computer is switched on. After location 1302H is set to 03, the motors may be engaged. The switch also comes in handy as a panic switch if your program causes the ET-2 to run amok!

Figures 3 and 4 show examples of ET-2 turning maneuvers. In Figure 3 the left motor is driven in reverse while the right motor runs forward, resulting in the preferred spin turn. In Figure 4 the right motor is driven forward with the left motor turned off, so that the left wheel is the axis of the turn, and the turn is more gradual. As mentioned above, the spin turn should be used for best results.

We will now describe how to reproduce these and more interesting movements using the SUPERKIM, both directly from the keyboard and then under program control.

Direct Command Mode

With the SUPERKIM interfaced to the ET-2 as previously described, constant motion modes can be commanded

TABLE 1

CONTROL LINE	FUNCTION (WHEN LOW)	J5 PIN
D1	RIGHT MOTOR ON	PIN 3 (A0)
D2	LEFT MOTOR ON	PIN 4 (A1)
E1	REVERSE RIGHT	PIN 11 (B0)
E2	REVERSE LEFT	PIN 12 (B1)

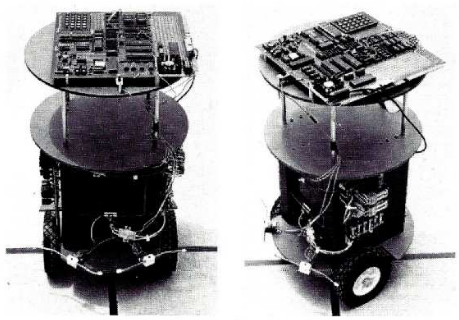


Figure 3. An on-axis turn. With one motor reversed, the ET-2 can turn in place.

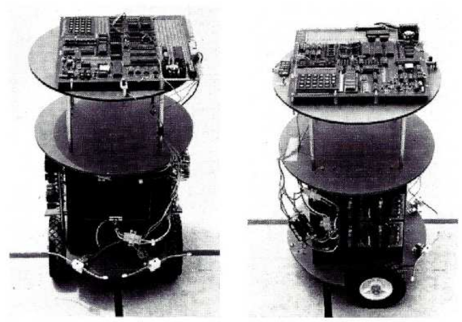


Figure 4. A "one-motor" turn. With one motor off, ET 2 turns with the stopped wheel as an axis.

directly from the keyboard as follows:

Step 1: Make sure that the motor switch is turned off (motor drivers disconnected from the computer) and then turn on the computer power switch. The display should light up.

Step 2: As described in the SUPERKIM manual, initialize the keyboard interrupt vectors as shown in Table 2. These values make the single step (SST) and stop (ST) keys work correctly.

Step 3: The ET-2 can now be commanded manually by entering the desired control states into address locations 1302H and 1303H. Table 3 shows the results of various output setings. *Note that the ET-2 should not be driven with both motors reversed, as the caster turns inwards and makes the unit unstable.*

Step 4: After the desired state is entered, turn the motor switch ON. *WARNING: In this mode the unit can only be stopped by turning the motor switch off, disconnecting the driver inputs from the computer!)*

Movement Under Program Control

While the direct command mode will allow you to check out your wiring, more complex sequences of control states must be commanded by machine language programming. Programs can be entered and debugged directly from the hexadecimal keypad on the SUPERKIM and then saved using the board's build-in cassette tape interface.

A highly desirable alternative to machine language programming is the use of a 6502 development system (APPLE, etc.). Instead of keying your program into memory in hex code, programs can be prepared on the development system using an assembler and then downloaded to the SUPERKIM through its serial interface. The advantages of using an automatic assembler to translate opcodes and compute the addresses for a new code file will become obvious the first time you have to add an instruction into the middle of an existing machine language procedure.

Table 4 is a listing of a 6502 machine language program for moving the ET-2 in a roughly octagonal pattern. It makes use of two nested time delay subroutines, LDELAY (long delay) at 0300H and SDELAY (short delay) at 0310H. SDELAY itself consists of two nested delay loops, each counting down from FFH to 0 (256 cycles) resulting in a delay of about 0.25 sec.

The byte at 0301H sets the loop count of the LDELAY subroutine, and is originally set to 2 as shown, for an aggregate delay of about half a second. Different delays may be obtained by using that byte as a subroutine parameter,

TABLE 2

ADDRESS	DATA
17FA	00
17FB	1C
17FE	00
17FF	1C

TABLE 3

ADDRESS	CONTENTS	CONTROL STATE
1302	00	BOTH MOTORS ON
	01	RIGHT MOTOR ON
	02	LEFT MOTOR ON
	03	BOTH MOTORS OFF
1303	00	BOTH RELAYS ON
	01	RIGHT RELAY ON
	02	LEFT RELAY ON
	03	BOTH RELAYS OFF

TABLE 4

ADDRESS	CONTENTS	LABEL	OPERATION	COMMENTS
0200	A9 03		LDA #$03	;POLYGON PROGRAM
0202	8D 03 13		STA $1303	;TURN RELAYS OFF
0205	A9 00	LOOP:	LDA #$00	
0207	8D 02 13		STA $1302	;BOTH MOTORS ON
020A	20 00 03		JSR LDELAY	;WAIT
020D	A9 03		LDA #$03	
020F	8D 02 13		STA $1302	;BOTH MOTORS OFF
0212	20 00 03		JSR LDELAY	;WAIT
0215	A9 01		LDA #$01	
0217	8D 02 13		STA $1302	;RIGHT MOTOR ON
021A	20 00 03		JSR LDELAY	;WAIT
021D	A9 03		LDA #$03	
021F	8D 02 13		STA $1302	;BOTH MOTORS OFF
0222	20 00 03		JSR LDELAY	;WAIT
0225	4C 00 02		JMP LOOP	;KEEP ON GOING
		;		
0300	A0 02	LDELAY:	LDY #$02	;SET DEFAULT COUNT
0302	8C 20 03	LOOP1:	STY COUNT	;SAVE IT
0305	20 10 03		JSR SDELAY	;CALL SHORT DELAY
0308	AC 20 03		LDY COUNT	;GET COUNT
030B	88		DEY	;COUNT DOWN 1
030C	D0 F4		BNE LOOP1	;CONTINUE TIL ZERO
030E	60		RTS	;RETURN
		;		
0310	A2 FF	SDELAY:	LDX #$FF	;OUTER CONSTANT
0312	A0 FF	LOOP2:	LDY #$FF	;INNER CONSTANT
0314	88	LOOP3:	DEY	;INNER COUNTDOWN
0315	D0 FD		BNE LOOP3	;LOOP UNTIL ZERO
0317	CA		DEX	;OUTER COUNTDOWN
0318	D0 F8		BNE LOOP2	;LOOP UNTIL ZERO
031A	60		RTS	;RETURN
		;		
0320	00	COUNT:	(long delay count hold location)	
			END	

setting it to a desired value "n" before calling LDELAY to give a total delay of n/4 sec. Finer control over the delay interval can be achieved by reducing the loop counts for the outer and inner loops within SDELAY (0311H and 0313H, respectively) from their original FF value.

The comments in the listing describe the action commands sent to the ET-2 at each step. This program makes use of the one-motor turn shown in Figure 4 (which may not be suitable for all surfaces). Since the outputs of the 6522 hold the values last set until the next output operation, the motor(s) will remain on (or off) during the call to LDELAY. The program has been simplified by using the default delay constant, 2, in the LDELAY loop. With just the right motor on, the robot will turn roughly 45° in the resulting interval, resulting in the approximate octagon pattern (Figure 5). Note that, as mentioned earlier, a power-off interval is commanded after each movement to minimize strain on the drive system (although it is not essential for a one-motor turn).

After the hex code in Table 4 is keyed into RAM at the locations give, the following steps should be followed to start the movement:

Step 1: Check the program carefully against the listing to verify each location. The single step (SST) button may be used to verify proper program execution (although stepping through the delay subroutines will prove tedious). Make sure that both the motor (1302H) and relay(1303H) output ports have been set to 03 (OFF). The motor switch may now be turned ON. Nothing should happen yet.

Step 2: Set the address to 0200, the start of the polygon program.

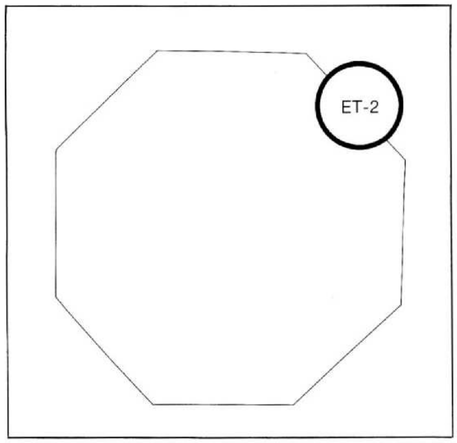


Figure 5. The (approximate) path resulting from the program.

Step 3: Press the GO button. The robot will begin to traverse an octagon.

Step 4: To stop the program, press the ST key and turn the motor switch off.

Stopping the program is best done during one of the pauses, when both motors are off. If the ST key is pressed while a motor is running, it (they) will remain running, due to the latching action of the 6522/6530.

Conclusions and Future Work

A more elegant method of obtaining the program delay would be to make use of the interval timer in the 6522. The device may be set to count up to 256 prescaled clock pulses by writing to the counter address. Based on the write address used to load the counter, the system clock will be divided by 1, 8, 64, or 1024 to produce the prescaled clock pulses. The unit will begin to count down at the prescaled rate as soon as a value is loaded. The register may be read by a program at any time to obtain the current count, and it may optionally be told to generate an interrupt upon reaching zero. Also, each 6522 has two 16-bit programmable counters, but these lack the ability to scale the count rate.

The SUPERKIM controlled ET-2 robot is an excellent, moderately priced system to which the robotics experimenter can easily add more sensors and other equipment. More elaborate systems may make use of the computer's versatile interrupt handling capabilities to design an event-driven real-time control system for the robot. Programs can also be written to use the 6522/6530 I/O ports for A-to-D conversion and interfacing the ET-2's contact sensors.

In the configuration described here, the computer controlled ET-2 falls short of the definition of a true robot, since all of its movements are "open loop." It has no sensors to tell it that a successful 45° turn has been made or even if it is travelling straight. Until the contact sensors furnished with the ET-2 are interfaced, even simple obstacle avoidance behavior is impossible. Part 2 of this article will describe the addition of sensors and interfaces to enable much more interesting behavior. A good source of additional 6502 machine language programs can be found in Tod Loofburrow's book. [4] These programs, with minor modifications, can be used for controlling the ET-2 with the SUPERKIM.

The ET-2 robot shell is well-built and reliable. The only problem I could find is that it has a tendency to tip over when being driven backwards at full speed with the rear caster in certain positions. Lour points this out and recommends that backing be avoided by doing a 180° on-axis turn instead. The unit can be driven over thick pile carpets without loss of traction, a task which many home robots find troublesome. Each motor draws around nine amps at full speed, so that the system needs recharging after an hour or so of continued use. Lour offers the unit in plan, kit, or assembled form.

References

[1] "KIM-1 User Manual," MOS Technology, 950 Rittishouse Road, Norristown, PA 19401 (August 1976).

[2] "Instructions for SUPERKIM," Microproducts, 2107 Artesia Blvd., Redondo Beach, CA 90278.

[3] "ET-2 Assembly Manual," Lour Control, 1822 Largo Court, Schaumberger, IL 60194.

[4] Tod Loofborrow, *How to Build a Computer Controlled Robot*, Hayden Books, Rochelle Park, New Jersey (1979).

Robots on your own time

Don McAllister
4709 Rockbluff Dr.
Rolling Hills Estates, CA 90274

SUPERKIM MEETS ET-2

PART II: SENSORS

In "SUPERKIM Meets ET-2" (*Robotics Age, Fall 1980*), I described how I interfaced and programmed a SUPERKIM single board computer (SBC) to control the Lour Control ET-2 robot shell.

Without sensors, though, the SUPERKIM/ET-2 combination described in that article is not a true robot, since all of its movements are "open loop," that is, without feedback. This article describes how to interface contact sensors and sensors that require A-to-D conversion (such as infrared scanners or temperature sensors) to the SUPERKIM/ET-2. Once you interface the contact sensors furnished with ET-2, you can program avoidance behavior. This permits the SUPERKIM/ET-2 to sense when it has contacted an obstacle, and take appropriate avoidance actions. I refer the reader to the Fall 1980 article for details concerning motion control of the ET-2 by the SUPERKIM.

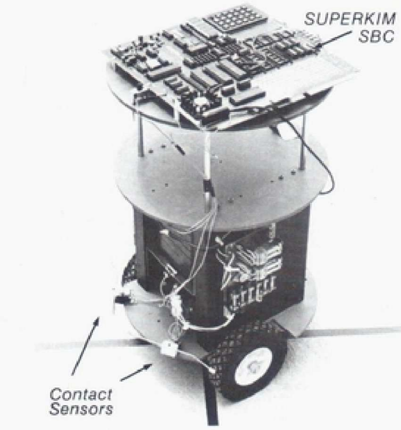


Figure 1. Contact sensors can be mounted around the base of ET-2.

Interfacing ET-2 Contact Sensors to the SUPERKIM

ET-2 provides a number of contact sensor switches that can easily be interfaced to the SUPERKIM. These contact sensors, equipped with metal "feelers," can be mounted around the base of the ET-2 to sense contact with an obstacle by means of a switch closure.

Lour Control has provided four independent contact bumper assemblies, which are designed to ring around the base of ET-2 as shown in Figure 2. Whenever a guard rod, which projects out of either side of the assembly, comes in contact with an object during ET-2's motion, it is deflected laterally, activating a built-in momentary switch. Depending on which way the switch is toggled, and on the control program in SUPERKIM, the ET-2 can then perform an avoidance manuever.

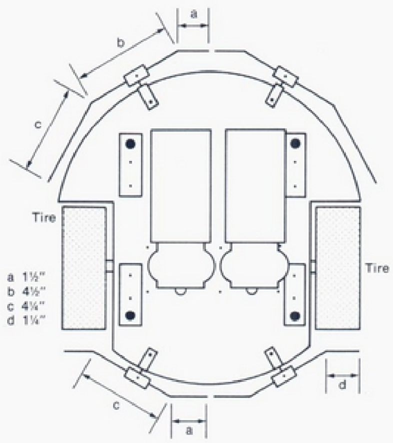


Figure 2. Location of contact switch/bumper assemblies.

As figure 3 shows, each of the bumper switches have four basic parts—the guard rod, connecting block, switch, and mounting bracket. The guard is a 5/32 inch diameter rod that protrudes from both sides of the connecting block and acts as an extension of the switch's own toggle lever. You can easily distinguish the two bumper assemblies installed in the front section of the shell, since their guard rods are shorter than those mounted in the rear section. The switch's toggle lever and the guard rod are both attached to the connecting block by means of set screws. The switch itself is a momentary, on-off-on device that automatically returns to the center (off) position when released. A spring wire, wrapped around the switch's mounting stud, holds the connecting block in a horizontal position and aids in the resetting of the switch. The entire unit is attached to one of the four mounting holes on the tier of ET-2 by means of a corner angle mounting bracket.

Table 1

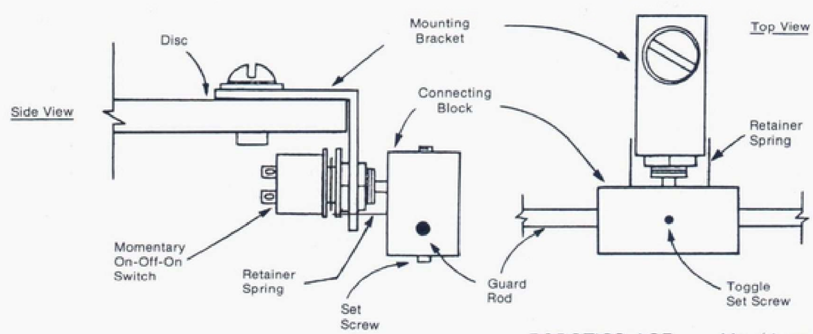
Connector Jack J2 (U11) 6530-003 I/O

Pin 0	=	Ground
Pin 1	=	PA0
Pin 2	=	PA1
Pin 3	=	PA2
Pin 4	=	PA3
Pin 5	=	PA4
Pin 6	=	PA5
Pin 7	=	PA6
Pin 8	=	PA7

We used SUPERKIM's 6522 to interface the SBC with ET-2's motor and relay controls. SUPERKIM also comes with two 6530 ROM/interface ICs, designated 002 and 003. To interface these sensors to SUPERKIM, we must first consider the operation of the I/O ports in a 6530. Each 6530 array provides 15 I/O pins. The microprocessor and operating program define whether a given pin is an input pin or output pin, determine what data are to appear on the output pins, and read the data appearing on the input pins. The I/O pins provided on 6530-002 are dedicated to interfacing with specific elements of the KIM-1 system, including the keyboard, display, TTY interface circuit, and cassette tape interface.

The I/O pins on the 6530-003 (U11) are brought out to connector jacks J2 and J3, and are available for user applications. Connector jack J2 has 8 pins constituting Port A, as shown in Table 1. Connector jack J3 has 5 pins constituting Port B, as shown in Table 2. Pin 0 on Port A is a ground line. Pins 1 through 8 on Port A and pins 1 through 5 on Port B are the programmable I/O lines. Figure 4 shows the location of the pins on the 6530-003 connectors J2 and J3 that are used as contact sensor input lines.

Figure 3. Contact sensor assembly (detail).



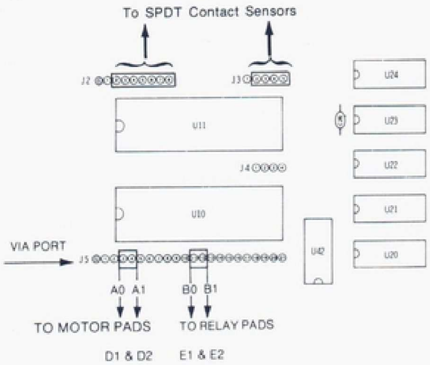


Figure 4. SUPERKIM/ET-2 contact sensor interface connections.

Each of the lines shown in Figure 4 go to one end of the desired (SPDT) contact sensor, as shown in Figure 5. Since the central pole of the switch is connected to ground, as the switch is opened and closed, the corresponding pin on jack J2 or J3 will be either an open circuit (corresponding to logic 1) or grounded (corresponding to logic 0). Read the data registers for Port A from memory location 1700H and the data registers for Port B from memory location 1702H.

You can interface the touch sensors by connecting one side of the SPDT switches mounted around the base of the ET-2 to signal ground and the other side to the appropriate pins of Port A and Port B. To understand how this connection works, consider the partial state diagram of the data register shown in Table 3.

If any of the pins PA1 through PA8 are connected to ground, then the corresponding state of the data line is set to zero, as shown in Table 3. The data byte stored in memory location 1700H—and read out by the KIM display—is the hexadecimal equivalent of the binary number represented by the states of the signals on PA1 through PA8, with PA1 being the least significant bit (LSB) and PA8 being the most significant bit (MSB). Thus, Port A alone can handle some 2^8=256 on-off contact sensor states.

Table 2

Connector Jack J3 (U11) 6530-003 I/O

Pin 1	= PB0
Pin 2	= PB1
Pin 3	= PB2
Pin 4	= PB3
Pin 5	= PB4

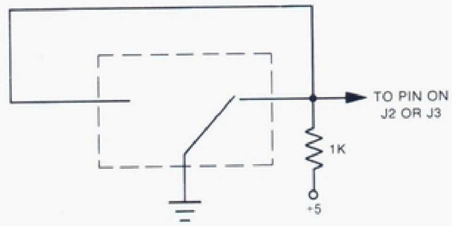


Figure 5. Contact sensor connection.

Motion Commands Based on Contact Sensor Data

The SUPERKIM can use contact sensor data to initiate a sequence of avoidance maneuvers any time the robot comes into contact with an obstacle. This behavior can be very complex, since a different avoidance maneuver routine can be triggered for every possible combination of contact sensor output. When all 8 contact sensors are mounted around the base of ET-2, the robot might use as many as 256 different avoidance maneuvers.

The principles behind this can be illustrated by considering two touch sensors on the front of ET-2, both wired to PA1 of Port A. In this case, KIM gets data byte FE if either front sensor contacts an obstacle. Table 4 gives a simple program making use of this data in a closed-loop fashion.

Execution of the program in Table 4 allows the SUPERKIM/ET-2 combination to go exploring somewhat in the manner of a billiard ball. The ET-2 moves forward in a stop-and-go fashion until one of the two forward contact sensors touch an obstacle. When this happens, the avoidance routine is called, which rotates SUPERKIM/ET-2 until the touch sensors are no longer in contact. Then the robot resumes its forward stop-and-go motion. Figure 6 shows the path of SUPERKIM/ET-2 under control of this program.

Table 3

Data Byte Equivalent of Port A Sensor Signals

Address	PA1	PA2	PA3	PA4	PA5	PA6	PA7	PA8	DATA BYTE
1700H	1	1	1	1	1	1	1	1	FF
	0	1	1	1	1	1	1	1	FE
	1	0	1	1	1	1	1	1	FD
	1	1	0	1	1	1	1	1	FB
1 = Open	1	1	1	0	1	1	1	1	F7
0 = Grounded (closed)	1	1	1	1	0	1	1	1	EF
	1	1	1	1	1	0	1	1	DF
	1	1	1	1	1	1	0	1	BF
	1	1	1	1	1	1	1	0	7F

As figure 6 shows, the path of ET-2 looks something like the trajectory of a billiard ball. By changing the program's delay constants at 0300H, 0311H and 0313H, you can change the angle of rotation of ET-2 during the avoidance manuever, as well as the duration of the start and stop motions.

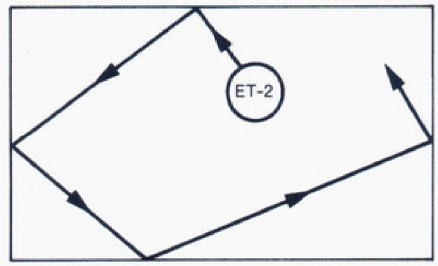


Figure 6. Path of ET-2 under control of the SUPERKIM "Billiard Ball" program.

Interfacing Analog Sensors to ET-2

Besides interfacing ET-2's contact (touch) sensors to the 6530 I/O parts you can also interface sensors that require analog to digital conversion (A/D). Sensors require A/D conversion when their output is a continuously variable signal or voltage as opposed to the 1 or 0 binary output of a touch sensor. Examples of such sensors useful in robotics are force/pressure transducers, temperature sensors, infrared sensors, or potentiometers used for shaft angle feedback in computerized servo control.

Table 4

"Billiard Ball" Program Listing

Address	Contents	Label	Operation	Comments
0200	A9 03	Loop:	LDA #$03	;Polygon Program
0202	8D 03 13		STA $1303	;Turn Relays Off
0205	A9 00		LDA #$00	
0207	8D 02 13		STA $1302	;Both Motors On
020A	20 00 03		JSR LDELAY	;Wait
020D	A9 03		LDA #$ 03	
020F	8D 02 13		STA $1302	;Both Motors Off
0212	20 00 03		JSR LDELAY	;Wait
0215	AD 00 17		LDA $1700	;Check Contact Sensor
0218	49 FE		EOR	;Compare with FE
021A	FO 03		BEQ (Z=1)	;Avoidance if FE
021D	4C 00 20		JMP LOOP	;Keep On Going
0220	A9 01	Avoidance:	LDA #$01	
0222	8D 03 13		STA $1303	;Right Relay On
0225	A9 00		LDA#$00	
0227	8D 02 13		STA $1302	;Both Motors On
023A	20 00 03		JSR LDELAY	;Wait
023D	A9 03		LDA #$03	
023F	8D 03 13		STA $1303	;Turn Relays Off
0242	A9 03		LDA #$03	
0244	8D 02 13		STA $1302	;Both Motors Off
0247	20 00 03		JSR LDELAY	;Wait
024A	60		RTS	;Loop:
0300	A0 01	LDELAY:	LDY #$01	;Set Default Count
0302	8C 20 03	LOOP1:	STY COUNT	;Save It
0305	20 10 03		JSR SDELAY	;Call Short Delay
0308	AC 20 03		LDY COUNT	;Get Count
030B	88		DEY	;Count Down 1
030C	D0 F4		BNE LOOP1	;Continue Til Zero
030E	60		RTS	;Return
0310	A2 FF	SDELAY:	LDX #$FF	;Outer Constant
0312	A0 FF	LOOP2:	LDY #$FF	;Inner Constant
0314	88	LOOP3:	DEY	;Inner Countdown
0315	D0 FD		BNE LOOP3	;Loop Until Zero
0317	CA		DEX	;Outer Countdown
0318	D0 F8		BNE LOOP2	;Loop Until Zero
031A	60		RTS	;Return From Subroutine
0320	00		COUNT: (Long	Delay Count Hold Location)
			END	

An LSI circuit, the ADC0817, is the primary IC in a 16 channel 8 bit A/D converter (ADC) system, which you can attach to the bus of the SUPERKIM 6502.* This ADC chip provides a relatively fast (100 microsecond) conversion time. Once the conversion has begun, the CPU can work on other tasks until the digital result is available.

The ADC0817 appears to the program as a block of memory starting at a base address, BASE, and extending through 16 locations to BASE + 15. (The actual circuit described occupies 4000 locations because of incomplete decoding which you can remedy if desired.) A conversion of a selected channel, say channel X, is started by writing to BASE + X. The 8 bit conversion result may then by read from any location in the block (eg. BASE) any time after the 100μs conversion time has elapsed. If you need multiple A/D conversions at the maximum speed, you can keep the 6502 busy with "housekeeping" during the conversion delay time. The system uses just five integrated circuits. The design, shown in Figure 7, occupies six square inches on the SUPERKIM prototype area, and draws only 60 mA of current from the 5 Volt DC power supply.

Operation of the circuit is simple because the ADC0817 performs all analog switching and A/D functions. The microprocessor R/W and ϕ1 lines, along with an inverted board select signal, are combined in two NOR gates, which 1) latch channel select bits A3-A0 and start A/D conversion during ϕ1 write cycles, and 2) enable the tri-state data bus drivers during ϕ1 read cycles.

You may want to take advantage of the SUPERKIM's interrupt circuitry to allow your program to go on to other tasks after starting the A/D conversion. The ADC0817 produces an end of conversion (EOC) signal when the most recent conversion has been completed. You can connect the EOC to a processor interrupt line (such as pin

*Both Texas Instruments and National Semiconductor produce the ADC0817.

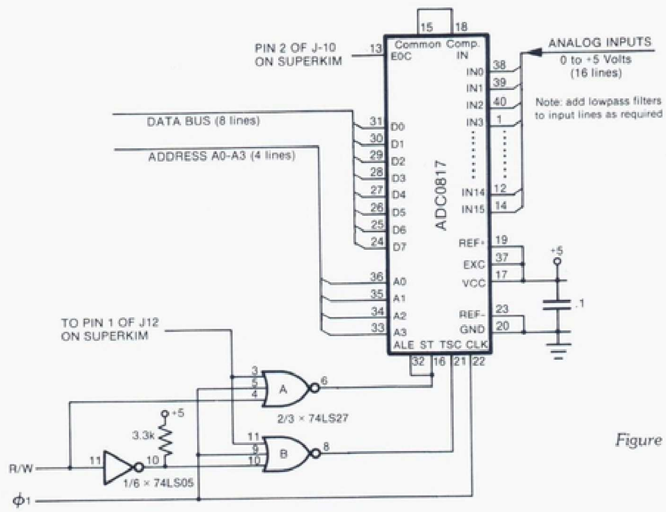


Figure 7. 16 channel analog-to-digital converter system.

2 of connecter J-10) through one of the 74LS05 open collector inverters. These interrupts can only be cleared by starting another A/D conversion. To use the interrupt feature, you must write additional software to initialize the processor interrupt and to "handle" the interrupt when EOC occurs.

Wire-wrap construction is suitable for the circuit—and component layout is not critical. It is good practice, however, to orient the analog input area away from digital circuits. The ADC circuit has two limitations: 1) analog input voltages must be between 0 and +5 Volts, and 2) the signals being converted should not change appreciably during the 100 μs conversion period.

Table 5 depicts subroutine MCAD for multi-channel A-to-D conversion without using interrupts, along with an example of a calling routine for MCAD.

The program which calls the A-to-D conversion subroutine must initialize both the channel selection and storage-defining parameters before the JSR instruction is executed. In the program given, the channel selection information is contained in an index register for ease of use in starting a conversion.

Table 5

A-to-D Conversion Routine

```
0200              BASE   *      $B000   ;BASE ADDRESS OF ADC0817
0200              STORE  *      $9000   ;START OF 16 BYTE STORAGE
                                               AREA
0200 9D 00 B0 MCAD   STAX   BASE    ;START CONVERSION ON
                                               CHANNEL X
0203 A0 0E               LDYIM  $0E     ;DELAY FOR CONVERSION
0205 88       DY         DEY            ;MINIMUM VALUE = $0E
0206 D0 FD               BNE    DY
0208 AD 00 80            LDA    BASE    ;GET CONVERTED DATA
020B 9D 00 90            STAX   STORE   ;STORE DATA
020E CA                  DEX
020F 10 EF               BPL    MCAD    ;DO NEXT CHANNEL
0211 60                  RTS            ;FINISHED
```

Example Calling Routine for MCAS

```
0212 A2 0F    MCMAIN LDXIM  $0E     ;SELECT CONVERSION OF ALL
0214 20 00 02            JSR    MCAD    ;16 CHANNELS AND GO TO
                                        ;SUBROUTINE
0217 00                  BRK            ;EXIT ** BE SURE TO INIT IRQ
                                        ;VECTOR**
```

Conclusions

The SUPERKIM controlled ET-2 robot is an excellent, moderately priced system to which the robotics experimenter can easily add more sensors and other equipment.

The contact sensors provided with the ET-2 leave something to be desired in that they do not make contact with overhanging obstacles such as tables and chairs. They do work adequately with vertical walls, and can be used to demonstrate obstacle avoidance behaviour in a suitably prepared environment. ■

References

[1] D. F. McAllister, "SUPERKIM Meets ET-2," *Robotics Age*, Fall 1980.

[2] "Instructions for SUPERKIM," Lamar Instruments, 2107 Artesia Blvd., Redondo Beach, Calif. 90278.

[3] "ET-2 Assembly Manual," Lour Control, 1822 Largo Crt., Schaumberger, Illinois 60194.