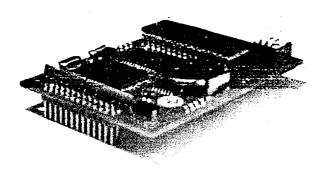
cpus

Scorpion K4S Controller

5520 and 5521



Introduction

The Scorpion K4S is the latest in the K4 range of credit-card-sized embedded control computers. Linked by a serial interface, the K4S may be programmed by virtually any computer running a terminal emulation program.

The K4 Language

Quickly understood by anyone familiar with BASIC, Pascal or C, the K4 language offers a powerful, easy to use multi-tasking environment. As well as procedures, functions and both integer and floating point arithmetic, a wide range of Object-Orientated drivers are provided to simplify commonly required user interface and I/O operations. A full screen text editor is also provided.

Control and Monitoring

Using the system bus and general purpose I/O channels, the K4S may be connected to up to 160 digital I/O channels and an unlimited number of analogue inputs and outputs. Selected digital channels may also be used for high speed pulse generation and analysis. The analogue channels can have 8 bit, 10-bit or 12-bit resolution.

Communication

Three serial output ports and two serial input ports are available, these may be used to connect the K4S to computers, instruments, printers, modems and other K4S controllers using our multi-drop serial protocol.

User Interface

The K4 language supports many types of both graphics and alphanumeric LCDs, as well as printers and a video display. The graphics LCD driver supports multiple windows, proportional fonts, word wrapping, justification and graphical operations, as well as 'active' bar charts and indicators. Combine this with support for multiple keypads (with up to 128 keys) the K4S can provide any application with a powerful user interface.

Application and Data Storage

The K4S comes with either 32K (the 5520) or 128K (the 5521) of RAM on board and has language support for non-volatile RAM files, compressed data buffers and fast access tables. Applications can be up to 128K in size and may be stored in either EPROM or FLASH memory.

5

The Scorpion K4 Language

Language Keywords

Control Flow

IF THEN ELSE WHILE DO UNTIL REPEAT FOREVER EVERY RETURN EXIT REGION CATCH THROW WAIT AWAIT

Operators

() . AS INT AS FLOAT NOT ? ! * / DIV MOD + - = <> >= <= AND OR EOR

Assignment

MAKE NEW :=

Printing

PRINT TO CR GOTOXY HOME CLS FONT LEFT RIGHT CENTRE PACKET DISPLAY COLOUR FILL ITALIC UNDERLINE PLAIN SIZE

Structure

TO [] END LOCAL

Multi-tasking

START STOP

System

HELP DELETE EDIT LIST PROGRAM CALL

Object Types

The key to the Scorpion K4S's rapid development cycle is the 'library' of user interface and I/O drivers built into the 'Object' system. This leaves you free to concentrate on your application without having to get involved in the details of writing hardware drivers.

Input/Output

Digital Analogue Shaft RateGenerator PulseWidthIn PulseWidthOut FrequencyIn PulsesIn

Communication

AsynchronousSerial InputBuffer OutputBuffer TerminalFilter Splitter ParallelPrinter I2Cbus

Timina

RealTimeClock Instant Monostable Stopwatch RunTimeLog

User Interface

GraphicsLCD Window AlphaLCD OnBoardLED Keypad Key Nudge Marker Bar Display NumberReader TeletextVideo

Data Storage

FilingSystem DataFile TextFile Buffer TextBuffer Table SafeData Box

System

K4 Debugger Task Semaphore

Speed and Size

The K4 language is interpreted, which means that programs take little memory. Things happen in the K4 language on a millisecond time scale. Since many commonly-required functions are implemented in the object system in machine code, the K4S can be surprisingly fast.

Example Application Program

What follows is a very short example of an application on the K4S: a farmer has a shed in which are stored bales of straw, and valuable stock is kept which must not be allowed to get cold. There are two conveyors continuously loading bales into and out of the shed. The K4S performs the following functions:

- Keeps a count of how many bales are currently in the shed
- Controls the shed's heating system
- Keeps a log of the environment and its own actions

Points to note

- Meaningful names are just as good as comments. Comments are declared with a semi-colon (;).
- Variables, routines and objects are all in lower case, eg. temp_reading, lcd display.
- K4S keywords are shown in upper case, eg. TO, END, IF, THEN.
- Object types and messages to objects are capitalised, eg. Buffer, AlphaLCD, Put, Asserted.
- A dot (.) means send the following message to the preceding object. It may return a result.

```
On power-on, the K4S looks for a routine called startup and runs it. The startup routine usually sets up the I/O configuration for the K4S application and
 TO startup
         ;** Run whenever the system starts **
            Standard MAKEs:
                                                                                                                                                                                                                                                                     declares some useful system objects
         MAKE system K4
        '.Read the switch to determine the baud rate
IF system . UserSwitch
MAKE serial AsynchronousSerial (1200)
ELSE MAKE serial AsynchronousSerial (9600)
         MAKE terminal TerminalFilter ("vt52", serial)
         system . TalkTo (terminal)
         MAKE led OnBoardLED
                                                                                                                                                                                           These MAKE statements create the objects that will perform all the I/O and some other system functions. An object is an independent piece of the K4S's system that knows have to perform some functions in response to a fixed set of messages. Note the SafeData object type; it stores data items in a place where a processor crash or power failure is very unlikely to corrupt it. Note also the Buffers. They may be configured to store only as many data bits as required for the particular use, even down to one bit oper entry.
         MAKE net 12Cbus
        MAKE clock RealTimeClock
MAKE tracer Debugger
MAKE ram FilingSystem ("RAM")
         :MAKEs specific to this application:
        MAKE lcd_display AlphaLCD (9 , 1)
MAKE temperature Analogue (256)
MAKE heater Digital (199)
MAKE opto_sensor_in Digital (200)
                                                                                                                                                                                                                                                                                              Here a Digital I/O object 'heater' is created to control the shed's
         MAKE opto_sensor_out Digital (204)
        MAKE safe_store SafeData (0)
bale_count := safe_store . Box (0)
MAKE temperature_log Buffer (1000 , 0 , 255)
MAKE heater_log Buffer (1000 , 0 , 1)
MAKE bale_log Buffer (1000 , 0 , 1000)
                                                                                                                                                                                                                                                                                               heating system.
        IF system . RunMode
[ ;** This part run only when in run mode **
    system . ErrorAction := 1
led . Flash (10)
                                                                                                                                                                                                                        If there is an error in the application program, then there are several options the K4S may take. This one means 'start the application again'
                 control_shed
END
 TO run
         system . Restart
 EMD
  TO control_shed
                                                                                                                                                                                                                                                                                     The highest-level routine is called after all the I/O has been set up. This then starts off the various portions of the control/monitoring system as independent tasks.
        START count_bales_in
START count_bales_out
START keep_a_log
         heating_control
TO count_bales_in
         FOREVER
[ PRINT TO lcd_display , GOTOXY (1 , 1) , "Bales:" bale count : 6
                 AWAIT opto_sensor_in . Asserted AWAIT opto_sensor_in . NotAsserted bale_count . IncrementBy (1)
                                                                                                                                                                                                                                                    These two routines are run in separate tasks, one counting bales coming in to the shed, the other counting bales out. Note that conflicts over which task has control over the bale_count value are taken care of by the monotonic update 'IncrementBy' and the automatic locking system on PRINTing makes sure that the alphanumeric LCD display is updated properly
END
 TO count_bales_out
        FOREVER
               PRINT TO lcd_display , GOTOXY (1 , 1) , "Bales:" ,
 bale count : 6
                 AWAIT opto_sensor_out . Asserted
AWAIT opto_sensor_out . NotAsserted
bale_count . IncrementBy (~ 1)
END
                                                                                                                                                                                                                                                                             This code is performed every 1000 milliseconds. Note the use of a local variable.
 TO heating_control
 LOCAL temp_reading
         | The control of the 
                                                                                                                                                                                                                                                                             Here the Digital I/O object 'heater' is
sent the message 'On'. This turns on
digital channel 199 (see where 'heater'
                THEN heater . Off
PRINT TO lcd_display , GOTOXY (1 , 2) . "Temp:" ,
                                                                                                                                                                                                                                                                             was created in 'startup')
PRINT 10 ...
temp_reading : 5 :
2 . " Heater is " . heater
END
TO keep_a_log

EVERY 3600 * 1000

{ temperature_log . Put (temperature)

heater_log . Put (heater . Value)

bale_log . Put (bale_count . Value)
                                                                                                                                                                                                                                                                                                         This routine keeps a log of what has happened in the shed, taking one set of readings every hour.
 END
```

Hardware

K45 I/O Channels

The bare K4S has 21 I/O 'channels'. Five of these are dedicated to the main serial port; the other 16 are general purpose. The table below lists the capabilities of these channels, each may take on one of the bulleted (•) functions at any time. Functions separated by an ampersand (&) can be used together.

Channel	Functions					
1-5	•RS232 Port 1					
6	RS232 Port 2 Data output (inv or non-inv TTL levels, up to 2400 baud, 1200 guaranteed)					
	•Pulse generation to 1.7 KHz, 0.4 μS resolution					
	•Digital I/O (pulled high 100KΩ)					
7	•RS232 Port 3 Data output (inv or non-inv TTL levels, up to 2400 baud, 1200 guaranteed)					
	•Pulse generation to 1.7KHz, 0.4 μS resolution					
	Keypad address line 2 & Alphanumeric LCD data line 6					
	•Digital I/O (pulled high 100KΩ)					
8	•RS232 Port 2 Data input (inv or non-inv TTL levels, up to 2400 baud, 1200 guaranteed)					
	•Pulse width measurement down to 286 μS, 0.4 μS resolution					
	•Pulse counting					
	•Frequency measurement to 3.4KHz, 20ppm precision					
	•Digital I/O (pulled high 100KΩ)					
9	•RS232 Port 2 Handshake input (inv or non-inv TTL levels)					
	•Alphanumeric LCD 'E'					
	•Digital I/O (pulled high 100KΩ)					
10	• 3S232 Port 3 Handshake input (inv or non-inv TTL levels)					
	•LTC1290 /CS					
	•Keypad address line 3 & Alphanumeric LCD data line 7					
	•Digital I/O (pulled high 100KΩ)					
11	•RS232 Port 2 Handshake output (inv or non-inv TTL levels)					
	•Alphanumeric LCD 'RS'					
	•Digital I/O (pulled high 100KΩ)					
12	•1 ² C Bus Clock					
	Digital I/O (not pulled up)					
13	•I ² C Bus Data					
	•Digital I/O (pulled high 10KΩ)					
14	•1 ² C Bus Data Direction (For extending the bus, not normally used)					
	•Digital I/O (not pulled up)					
15	•Keypad return line					
	•LTC1290 /CS					
	•Digital I/O (pulled high 100KΩ)					
16	•On board LED					
	•Digital I/O (pulled high by an LED and 3K3Ω)					
17	•NMI (Non-Maskable Interrupt line - not currently used by the language, but available to machine code					
	users, pulled high by $100K\Omega$)					
18	•IRQ (Interrupt request - not currently used by the language, but available to machine code users)					
	•LTC1290 Dout					
	•Digital I/O (pulled high 100KΩ)					
19	•Memory Ready - used by devices attached to the System Bus					
	•LTC1290 /CS					
	•Digital I/O (pulled high 100KΩ)					
20	• Keypad address line 0 & AlphanumericLCD data line 4 & LTC1290 Din					
	•Digital I/O (pulled high 100KΩ) •Voymed address line 1 & Alphanumeric LCD data line 5 & LTC1290 SCLK					
2 1	Reypad address life i d infinitelities					
	•Digital I/O (pulled high 100KΩ)					

Expanding the K4S

Many more functions may be added to the K4S via the I^2C bus, Expansion bus, UPP and the LTC1290. These are described below.

I²C Bus

The I²C bus (pronounced 'I Squared C') is a two-wire network which can control many devices. On the K4S, the I²C bus makes available the following functions, which are supported by the language, but are not part of the basic K4S hardware.

• 128 digital I/O channels.

- 32 analogue inputs channels and 8 analogue output channels (8 bit resolution).
- A real time clock/calendar.
- A teletext video display.
- The language also supports a parallel printer, alphanumeric LCDs, and keypads, all based on I²C Digital I/O channels.

In addition, the language provides direct access to the I²C bus so that unsupported devices may be controlled.

System Bus

This is the K4S's raw processor bus, running at ~2.4 MHz, to which the UPP, Expansion Bus and other devices may be attached.

Expansion Bus

This is a software-controlled I/O expansion port made by the addition of a few cheap ICs, attached to the system bus. It allows a graphics LCD and large keypad to be attached to the K4S. It will also drive alphanumeric LCDs.

UPP

The HD63140 UPP (Universal Pulse Processor) is an IC which may be attached to the K4S's System Bus. On the K4S, the UPP makes available the following functions which are supported by the language but are not part of the basic K4S hardware.

- Up to 16 pulse counting inputs (100KHz max)
- Up to 8 shaft encoder inputs (200KHz max)
- Up to 6 frequency measurement inputs (100KHz max, 25 ppm precision)
- Up to 6 pulse width measurement inputs (5μS resolution)
- Up to 11 Pulse generation outputs (5µS resolution)
- 10 analogue inputs (10 bit resolution)
- Up to 16 digital I/O channels.

LTC1290

The LTC1290 is a 12-bit analogue-to-digital convertor IC that may be attached to the K4S using the general I/O lines. The LTC1290 is supported by the K4 language's Analogue object, but does not form part of the basic K4S hardware. Up to 3 LTC1290 ICs may be connected at any one time, each providing 8 analogue inputs with 12-bit resolution.

Real Time Clock/Calendar

The K4S has a socket for a real time clock/calendar IC. The language supports this device over dates ranging from 00:00:00 on January 1st 1990 until the year 2024, taking account of leap years. The K4S's on-board battery keeps the clock's date and time correct even when the unit is unpowered.

Watchdog

The K4S has an on-board watchdog circuit that will normally restart the K4S's application program if the processor 'crashes'.

Characteristics

Absolute Maximum Ratings

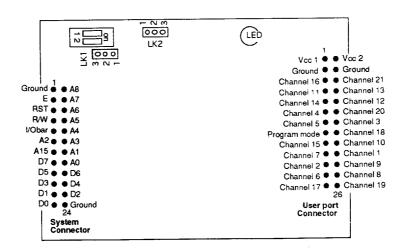
Quantity	Symbol	Min	Max	Unit	
Supply Voltage	Vcc	-0.3	+7.0	V	
Operating Temperature	Topr	0	70	°C	
Storage Temperature	Tstg	-55	150	°C	
Input Voltage (all lines)	Vin	Vss-0.3	Vcc+0.3	V	

Electrical Characteristics

Quantity	Symbol	Test condition	Min	Тур	Max	Unit
Supply Voltage	Vcc		4.6		5.5	V
Operating Current	Iin			25		mA
Doze Mode Current	Id			6		mA
Standby Current (Sleep Mode)	Is			10		μΑ
Output High Voltage	Voh	$Ioh = -200\mu A$	2.4			V
Output Low voltage	Vol	Iol = 1.6mA			0.4	V
Input Leakage Current*	Iin	Vin = 0.5 - Vcc - 0.5 V			1.0	μΑ
Input "High" Voltage	Vih		2.0			V
Input "Low" Voltage	Vil				0.8	V
Darlington Drive** (All onboard channels except 3,4,5,18,19,20)	-Ioh	Vout = 1.5V	1.0		10.0	mA

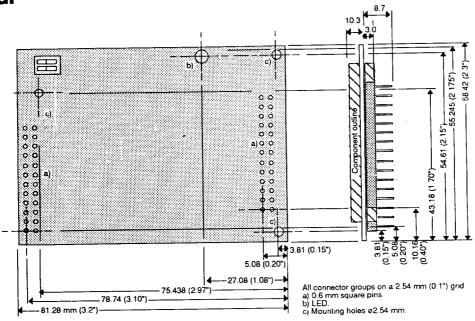
^{*}This must be modified by the fact that all channels are pulled to Vcc by $100K\Omega$, except channels 5, 12 and 14 which are not pulled at all, channel 13 which is pulled high by $10K\Omega$ and channel 16 which is pulled high by an LED and $3K5\Omega$.

Pinout



^{**}These outputs can drive NPN darlington transistor bases directly.

Mechanical



Weight:

36 grams

(46 grams with EPROM and Real Time clock chip)

Dimensions:

82mm by 59mm by 24mm

Getting started

To start evaluating the K4S you will need:

- A computer emulating a VT100 or VT52 terminal at 1200 or 9600 baud
- A K4S controller
- A K4 Language EPROM or FLASH language kit
- A K4 Application board
- A suitable power supply
- K4 Introductory Manuals

See the 'Entry Packs' section for information on getting started with K4S.