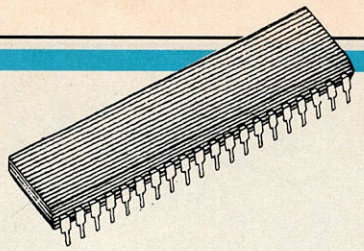


MICROCOMPUTERS

Single chip microcomputers using FORTH

Two single chip, FORTH based microcomputers have been produced by Rockwell International. These have a FORTH kernel in an on-chip 3K byte ROM. The two devices are called the *R65F11* and the *R65F12*, and represent substantial computing power in their 40 and 64 pin packages.

Available in 1 or 2MHz versions, each device contains the new enhanced *6502* CPU, with 4 extra instructions. These are special bit manipulation instructions, Set memory bit, Reset memory bit, Branch on Bit Set and Branch on Bit Reset. These commands save using the accumulator for setting individual I/O bits (as is often required with a VIA device for example). There are 192 bytes of RAM, 32 bytes of which can be held on standby power when current to the rest of the device is switched off, two 16 bit timer counters and 16 I/O lines on the *R65F11*, or 40 I/O lines on the *R65F12*. Up to 16K of memory can be addressed, although in development configurations, there are methods of using the Rockwell FORTH system with a 64K memory map.

Development mode

The devices work in two modes—Development mode and Target mode. In Development mode, the *R65F11* or *R65F12* will be used in conjunction with the *R65FR1* development ROM, and a RAM device. Contained in the *R65FR1* ROM are terminal and I/O support features for disk drives and printer, as well as FORTH words to enable generation of target compiled code for use in a final application.

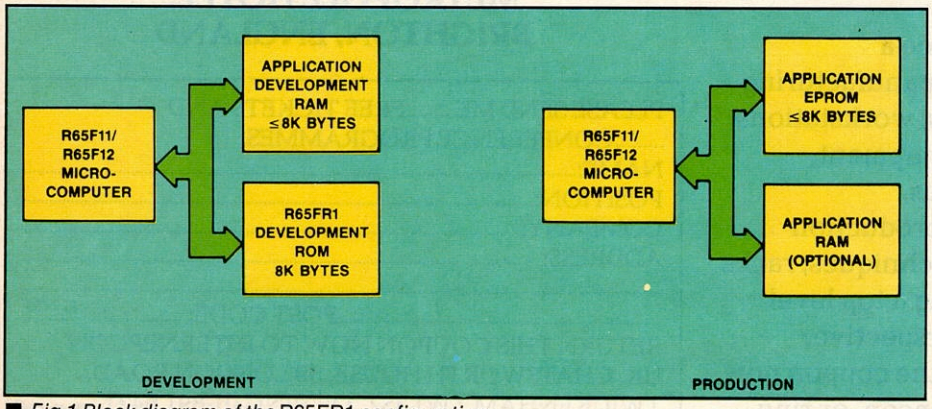
If disk drives are being used, then the device supports a standard screen editor for FORTH source code using the disk drives as virtual memory storage. Alternatively, a host computer with a simple word processor and RS232 facilities can be utilised. Preferably such a host system should also be capable of terminal emulation for use when debugging.

Software debugging

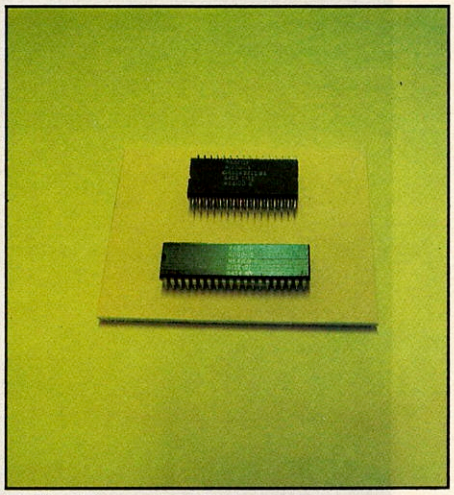
One of the greatest benefits with these devices is that software can be fully debugged in an interactive manner on the target system. By downloading FORTH source code in ASCII to the *R65F11* or *R65F12*, and then operating the computer via a terminal (or terminal emulator program on the host computer), each FORTH word in the target application can be individually and interactively tested.

Two boards are currently available, which can be used to develop RSC FORTH code, or as target systems.

A single board computer from RCS Microsystems is designed as a complete development system for RSC FORTH. The board contains the *R65F11* processor, 6k of RAM, the *R65FR1* development ROM, and a disk controller. Add a 5 volt power supply, terminal and disk drives (up to four 800k drives) and RSC FORTH object code can be produced in a FORTH screen environment ready for introduction into a target system. EEROMS can be programmed on the board itself, or the code can be downloaded to an RS232 type PROM programmer such as the *STACK S2000*. This module, with disk drives, provides the ideal system for developing FORTH code, as the editor, storage and facilities for debugging are all present in the one system.



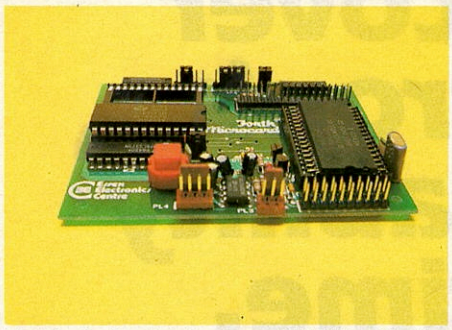
■ *Fig.1 Block diagram of the* R65FR1 *configuration*



■ *Fig.2 The two FORTH based microcomputers —designated the* R65F11 *and the* R65F12*—are available in 1 or 2MHz versions. Each device contains the new enhanced* 6502 *CPU*

The FORTH microcard

The FORTH microcard packs a lot of power into its small size. The board measures just 80mm × 100mm, and contains two 28 pin sockets, and an *R65F12* microcomputer. Configurations are development mode, plus three



■ *Fig.3 Containing an* R65F12 *μC, the FORTH microcard board measures 80mm × 100mm*

target modes giving various amounts of ROM and RAM (up to a total of 15k external memory). All of these configurations are selected by the use of minijumps.

This board can be used both as a development system using a host computer as described, or as a target system. The board provides 40 lines of I/O including an RS232 interface.

For those with slightly different requirements of a target system, or needing more than 16k of memory available, the *R65FK2* and *R65FK3* FORTH kernel ROMs can be used. The *R65FK2*, used in conjunction with a *R6501Q* single chip microprocessor emulates the FORTH processors. Of more interest, however, is the *R65FK3* system which again uses the *R6501Q* to provide a massive 64k FORTH system. Once more, development takes place on the target system.

Target mode

Once the target program has been fully written and debugged, it can be transferred into ROM and operated in 'Target Mode'. This has several advantages:

- The code can be generated without the ASCII FORTH headers, thereby using up less memory space.
- The *R65FR1*, -2 or -3 development ROM is not needed, so a full 16k (less 256 bytes) is available for programs and data.
- The program is set to start automatically on power-up, and is also therefore secure.
- The 'operating system' is inside the FORTH microcomputer itself, with only user program externally.

Programming

Certain features have been added to enable full use of the single-chip microcomputer environment. A FORTH assembler is included in the development ROM and this makes the inclusion of time-critical sections of code simplicity itself.

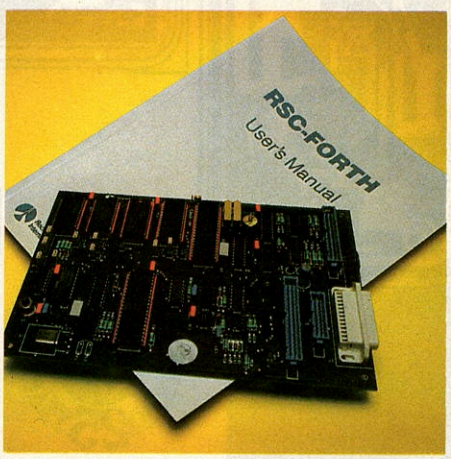
Also included are words which add autostart vectors, generate headerless code and other functions needed to target a FORTH program. Interrupts can be handled either in native *6502* machine code, or with high-level FORTH code. These interrupts can be generated from edge-sensitive inputs, serial input data register full, serial output register empty, or from the 16 bit timers counting down to zero from a preset value. This is probably the most used type of interrupt, and is used for interval timing, or for real-time clock generation.

The RSC-FORTH manual supplied with the *R65FR1* ROM gives two listings of a 24-hour clock program. The first uses machine-code interrupts, and the second demonstrates the use of FORTH to handle 100 interrupts a second.

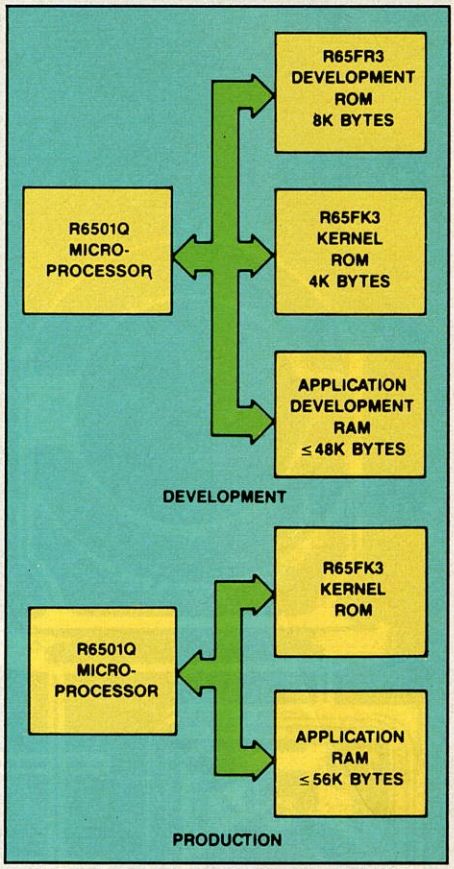
Generating target code

The process for generating target code is as follows:

1) Write and debug the program in the target system.
2) Ensure that all variables that are required to be in RAM are separated from the main program. (This will be put into ROM.)
3) Set the dictionary pointer to 4 bytes past a 1k boundary with the use of DP ! (eg 0804 DP !).
4) Instruct the FORTH operating system to generate separate code and header fields with H/C. It is normally best to put these into high RAM, (eg 1000 H/C).
5) Re-compile the code into RAM.
6) Set the system to autostart by using the 1k boundary chosen as a parameter for AUTOSTART. (eg 0800 AUTOSTART PROG), where PROG is the name of the application.
7) Dump the RAM image to a PROM programmer or transfer it using an EPROM emulator.
8) Generate the new ROM program.
9) Re-install the PROM and press RESET.



■ *Fig.4 The RSC-FORTH development module*



■ *Fig.5 Block diagrams for the* R65FR3 *and* R65FK3 *configurations*

The system should come up and running with the target application in control.

In using FORTH as the programming language, benefits such as reduced development time, easier software maintenance and enhancement, development of software in the target system as well as straightforward generation of target code, Rockwell have produced a versatile range of single chip microcomputers. These devices will surely find themselves being used in many and varied applications for years to come, and are already being adopted by industry as a standard component.

Simon Taylor
RCS Microsystems