

WRITE - BACK

ON ENTRY

Those earlier magazines of yours are absolute gems. I ask in all humility if you have ever thought of sorting them all out into various topics and re-issuing them in the form of booklets which I am sure would be very popular. I seem to spend an extraordinary amount of time searching for information through various books and magazines often unsuccessfully. Because they were written so long ago it is sometimes difficult to judge whether the methods and information you gave then have been overtaken by the rapid advance in technology (those VDU commands for example).

My main interest is animation, and it was in an effort to get to terms with user sprites that I got involved in the first place with assembly language. I found a couple of articles on user sprites and several on assembly language programming in your magazine most helpful. All the same a favourite ending is "For more information consult the Programmer's Reference Manual".

Well I have inherited a pretty ancient copy of this hallowed manual true, but so far have had no incentive whatever to update it.

Just about any topic I try to consult it on starts off "On entry R0=0" or 1 or 2 or whatever. Total defeat.

What in heaven's name is R0? And what does it mean by "On entry" let alone by "On exit"? And how do I get it to equal 0 or whatever? Try as I might nowhere can I find an explanation of this mysterious command.

You might ask how I got involved in assembly language if I don't understand this basic concept, and my answer can be that it only illustrates the enormous and growing gap between those who can and those who can't. I have a copy of Mike Ginns book Archimedes Assembly Language which is excellent as far as later chapters go. I can AND and EOR and Two's Complement with the best of you, and I know all about condition codes and branching with links etc. and can even write simple programs of my own, but every time I think I understand the first couple of chapters, I only need to pick up the Programmer's Reference Manual to find I don't.

Thelma Webb

Much of what was published in earlier issues of RISC User is still as valid now as when it was first published. However, volume 1 was published before the release of RISC OS, and in

that respect (but that only) these early issues are less useful. Acorn has always put considerable emphasis on upward compatibility, and this is still as true today with the new Risc PC range.

References to R0, R1 and the like are to the registers in the ARM processor. In Basic, references to these locations are an intrinsic part of SYS calls (the SWI call being the direct equivalent in assembler). As an example, the following statement will result in the value of 65 being placed in R0 when the SYS call is made:

```
SYS "OS_WriteC",65
```

While there are few books on

ARM POWERED

programming the Archimedes, a recent one is First Steps in Programming Acorn RISC OS Computers by Martyn Fox, published by Sigma Press at £14.95, and this has a chapter on SWI calls and assembly language. The book Wimp Programming for All by Lee Calcraft and Alan Wrigley published by RISC Developments at £12.95 also contains much useful and relevant information on SWI calls from a Basic programmer's point of view, though the emphasis is on writing Wimp programs in general rather than on graphics or animation.

Despite two excellent articles by David Spencer (RISC User 5:4 and 7:3) on the ARM600, ARM700 and

beyond, it is still not clear to me what their connection, if any, is with Apple Computer Inc and the new PowerPC chips.

Apple are shareholders in ARM Ltd and there have been various hints that Apple's Newton is powered by an ARM6 derivative. Contrary to this, Apple's new PowerPC 601 RISC 32/64 bit transitional chip is said to be the product of Apple, Motorola and IBM, with no mention of any British contribution. Both Apple and IBM intend to produce computers based on the Motorola PowerPC 600 series, presently ending with the PowerPC 620, a true 64 bit version of the 601; with a 700 series in the very distant future.

Meanwhile Acorn is reported to be bypassing the 600 series and concentrating on the ARM700. Yet David Spencer concludes his article with "The ARM7 series is really only a modest progression

PIN BOARD

MAC-IN-DOS

With reference to the article on pages 57-58 of RISC User 7:5 (The PC Emulator Survival Guide) about MAC-IN-DOS, I've just downloaded the HENSA index for micros/ibmpc/dos and unfortunately MAC-IN-DOS has been removed - a pity as it looks to be a useful program. Further it now appears that HENSA can be accessed direct on 0524 843878 using a scrolling terminal set to 8N1 at speeds up to 2400.

Gareth Elderkin

If readers know of any other source for MAC-IN-DOS, perhaps

from the ARM6 family of devices". From the little information I have seen, Apple and IBM's transitional 601 chip, with its 32K cache and integral FPU, appears more powerful than the ARM700, not to mention the 64-bit 604 and 620 chips now being tested. Is this the case, or am I missing something? Is there any connection between the ARM600 and the PowerPC 600 series. Whatever the

answers, the future certainly looks very bright for RISC computing; let's hope that Acorn, who pioneered RISC technology, do not miss the boat.

Peter Ghiringhelli

First of all there is no direct connection between the ARM processor and the PowerPC chip, apart from the fact that both designs



Points Arising.....

In our review of the Observess expert system shell (RISC User 7:7) we inadvertently omitted the product details. These are as follows:

Product	Observess
Supplier	Cherisha Software
	51 Swallowfield Road
	Charlton
	London SE7 7NT
	Tel. 0956 389500
Price	£20 (no VAT)
	(Site licence £60)

We apologise for the omission.

Version 2.00 of Observess should be available by the time you read this.

AUTOSTEREOGRAMS - RISC USER 7:7

Unfortunately the gremlins crept into the listing accompanying this article. Everything was correct when the article left us on its way to the imagesetter, but somehow the formatting of the listing went awry in the process. The result was that several lines have gaps in the wrong places, and more seriously, 5 lines went missing from the end of the program. The missing lines are as follows:

```

300 OSCLI("SetType "+outfile$+" FF9")
310 END
320 DEF FNget:ptr%+=1:=?ptr%
330 DEF FN3D
340 IF flag%=0 D%=FNget:flag%=1:=base-
(D%AND&0F)
350 flag%=0:=base-(D%AND&F0)/16

```

You should be able to work out how the lines