New electronic memory chips, a new microprocessor chip, the UART defined & some circuits, how drawings are made

By Don Lancaster

We’ll once again start out with our usual reminder that this is your column and you can get solutions to many of your “Hardware Hacker” questions per the phone and address in the ending box.

Since there’s lots of exciting new hacker components this month, let’s just jump right in.

What’s new in electronic memory?

So much is happening so fast it is hard to keep up. Let’s look at three new developments:

First, if you check the ads right here in Modern Electronics, you will find 256K dynamic RAMs available for a mere $3 each, give or take a few cents. That’s almost one millicent per bit!

What utterly boggles the mind is that if “they” keep increasing memory size and keep reducing the per-bit price at the same rate “they” have for the past 14 years, you will have the equivalent of the 5-billion neurons in the human brain on a single chip for under a dollar before the end of the century!

Second, Apple has a new RAM card that holds a full megabyte of memory and fits either a II+ or IIe. The card automatically supports both the DOS 3.3 and ProDOS operating systems. You can use as many cards as you need, up to a total of seven or eight megabytes. Plus the 64K or 128K you started out with.

I’ve started using this card and I like it very much. It is destined to become the primary standard for Apple add-on memory cards. Why? Because it (a) comes from the “company,” and (b) does not use awkward and ungainly bank switching. Instead, the entire card is readable from one single address that auto-increments.

I have put together a disassembly script and a few utilities that make this card more understandable and far easier to use. Write or call if you need more on this.

Third, Both NEC and OKI have some new memory modules out that show outstanding potential for the advanced hardware hacker.

As Fig. 1 shows, these modules consist of a fancy printed-circuit board and nine 256K dynamic RAMs in a small plug-in unit. The ninth RAM can be used either for error testing or as a spare that gets jumpered if another RAM chip fails.

Besides being much smaller than traditional DIP packaging, these new modules are removable and contain internal supply bypass capacitors. They are almost certain to become the RAM of choice for the next generation of microcomputers.

OKI will sell you two 256K x 1 modules and a special double socket for only $48 in singles. That’s right at the magic millicent per bit figure. Its called the DRAM SIMM Sample Set, and is available right now. Before you get all excited, though, note that these are strictly for advanced hackers only. There are serious hassles involved in using any dynamic RAM. You have to know where the address lines are going to come from to address an entire 256K block of words. You have to know how you are going to refresh the RAM chips to keep the memory alive. And, you have to know how to handle address pin multiplexing where one package pin is used at different times for either a column address or a row address.

Nonetheless, these RAM modules should drop into a Macintosh with a minimum of hassles or rework. More information on dynamic RAM appears in my newly reprinted Micro Cookbook, Volume I (SAMS #21828).

Any new micros this month?

I just ran into a really exciting new microprocessor chip that should be ideal for almost any small controller, trainer, or data gathering use. This chip has hacker written all over it.

It’s called the Mitsubishi M50734 and sells for $12 in single. It is built-in CMOS and speaks 6502, so it should be really easy to use with Apples, Commodores, Ataris, and the rest of the 6502 gang. Internal bank switching is used to create a double-sized 128K address space.

No RAM or ROM inside, but my oh my, the ready-for-peripheral circuitry stuffed into this beast. How does 40 I/O lines, a Centronics port, nine timers, a multiple input A/D converter, some step-
per motor drivers, a pulse position modulator, and a two-way UART serial communications channel interface sound? If that's not enough for you, there are enough pins on the package that you can also use it for an emergency cheese grater.

Obvious uses for the M50734 include data recorders, hydrophonic garden timers, cable TV intelligent taps, solar-powered pump interfaces, general robotics, weight scales, cattletender setups, student trainers, hot-tub pump cyclers, cotton-picker tooth setter, numeric machine controls, etc.

It might even be possible to drop this dude and an adapter directly into the 65C02 socket on an Apple IIC or IIE and run everything else out of a humongous new I/O connector. Stay tuned to this column for more details.

The beauty of being more or less Apple compatible is that the full resources of the Apple become available for design, debugging, and testing of what you want to do. No costly ROM masks or emulation software is needed, and the design process is fast and fun by using standard assemblers, debuggers, printers, disk drives, etc.

Two minor gochas. The 64-pin, 70-mil shrink DIP package is oddball, so sockets may be very difficult to find. In addition, there are not enough pins to go around, so they used Motorola style pin multiplexing on the data/low address lines. At worst, this means you have to add a 74HC373 address latch to your final circuit.

You'll find lots of information on programming 6502-style chips like the M50734 in my two Micro Cookbooks, SAMS #2128 and 21829.

Tellyawhat. Let's have us another contest. A free SAMS book to the best 10 uses for the M50734. The overall winner gets an all expense paid tinaja quest for two (FOB Thatcher, Arizona), plus some possible cash type money if their entry is good enough to become a Modern Electronics feature or construction article.

What is a UART

There are two different ways of sending data between two points. With parallel data communications, all bits in a data word are simultaneously sent using as many lines as there are bits. Parallel data communication is very fast but need lots of wires. On the other hand, serial data communication needs only a single wire to get from point A to point B. All bits are sent down the line one at a time in a specific order. Serial data communication is much slower than parallel, but it needs only a single channel.

In general, parallel data communication is used inside or near computers, while serial data links are used between computer systems, particularly over long distances. Consider the absurdity of needing eight separate telephones at both ends to send 8-bit data words.

There are lots of times when you want to convert from parallel data to serial or vice versa. For instance, a lap or handicapped keyboard would best have its data sent to a computer in serial form. Some computers, such as the Apple IIC, output data only in serial form. You might like to convert to parallel outside the computer to drive a parallel dot-matrix printer or a power controller full of relays or solid-state power switches.

Some electronic circuits do not have a microprocessor in them. Except for the simplest of computers, the main microprocessor in a microcomputer cannot be expected to sit around all day to generate slow serial code or wait until new serial input data arrives. The micro almost always has better things to do. Instead, there are many special ICs that convert data from serial to parallel and back again as needed. They are cheap and easy to get.

These special circuits are called UARTs, short for Universal Asynchronous Receiver Transmitters. The "asynchronous" in the name means that any amount of time can go by between successive transmissions or receptions.

There are two basic types of UARTs. A hardwire UART is used when there is no microcomputer involved at its end of the communication channel. Two examples of hardware UARTs are the Intersil IM6402 and IM6403. With hardwire UARTs, the data formats are set by physical jumpers, and the input and output data appears on individual lines.

A Peripheral style UART is intended for use with a controlling microprocessor. The Rockwell 6551 is a typical example. With a peripheral UART, the data format and sometimes the data rate is settable by software. Input and output parallel data is usually routed directly to the data bus on the controlling micro.

Figure 2 shows the usual format of a serial asynchronous code. This was originally called the Teletype code. Very often this code is sent from system to system by using RS-232C standard levels and pinouts. Note that a code and a standard are two totally different things.

You always start with a start bit that tells the receiving UART that a new character is to arrive. Remember that with asynchronous communication, a random amount of time can go by between individual transmissions. The actual code bits then follow in sequential order, starting with the LSB, or least-significant bit. After the code bits a minimum of one or two stop bits is sent, followed by as many more stop bits as you need to wait until the next transmission.

The speed of transmission is set by the baud rate. The baud rate is defined as how fast the bits come out of the pipe. Popular

![Fig. 2. The standard 8-bit asynchronous serial code.](https://example.com/fig2.png)
HARDWARE HACKER...

baud rates include 300 baud for modems and 9600 baud for printers. You can roughly relate baud rate to cps (characters per second) by dividing by 10. Thus, a 300-baud modem link maxes out at 30 cps.

Most UART circuits need input reference frequencies that are far higher than the communication baud rate. Among other reasons, this lets the receiver circuit sample only the very center of a received data bit for minimum noise. Often, but not always, the system baud rate reference will be 16 times the actual transmission and reception baud rate.

While eight data bits are usually the norm, five or seven are sometimes used. It is also possible to use an optional error-checking parity bit, although this is not done too much in local microcomputer use. The parity bit forces all of the ones in a word to an even number; if an odd number was received there was an error. Odd parity can also be selected.

It is extremely important that the baud rate, number of data words, number of stop bits, and parity (if any) agree between sending and receiving ends. Since both baud rates must agree to better than one percent accuracy, crystal control is normally used at both ends.

Show me some UART circuits.

The Intersil IM6402 is a good choice for a modern hardware UART. This one is available from Digi-Key $6.90, (part no. NT5000-ND) among other dealers. The IM6402 is pin-compatible with earlier UARTs, but needs only a single +5-volt power supply.

As with earlier hardware UARTs, the IM6402 needs an external 16 x baud rate that is input to the transmitter on pin 40 and input to the receiver on pin 17.

The number of data, parity, and stop bits is programmed by jumpering pins 34 through 39 to ground or +5 volts. Making all of these +5 volts gives you eight data bits, no parity, and two stop bits. Consult the data sheet for any of the other combinations if you need them.

Naturally, you should never use any integrated circuit without the data sheet and any available ap notes on hand and these should be thoroughly understood before you do use a given UART.

In general, the "top half" of a hardware UART is used to transmit, and the "bottom half" to receive.

The IM6403 is a variation on the IM6402 that has some interesting advantages. Instead of a pair of 16 x baud rate inputs, you can hang a crystal directly on the chip and generate your own internal baud rate. Unfortunately, many popular baud rates need higher-frequency crystals that require higher supply voltages, preferably +9 or +10 volts, or the baud rate generator will not oscillate reliably.

Figure 2 shows a UART transmitter that is set to 4800 baud, uses a cheap crystal, and runs off a +5-volt supply. A transmitter accepts parallel data in and outputs serial data. Pins 34 through 39 are shown programmed for eight data bits, no parity, and two stop bits. The inverter on the right
is one way to do a power-on reset. Any other scheme that briefly brings pin 21 high after power-up will work as well.

One possible use for this circuit is a lap or handicapped keyboard.

Figure 4 shows a UART receiver that accepts serial input data and outputs parallel words. One obvious use would be for a dot-matrix printer adapter for the Apple IIc personal computer.

That bottom inverter requires some careful explanation. In most fancy UART systems, you need full handshaking to make sure that each received character gets used once and only once. To do this, the DR or data ready pin goes high when data is received. When the rest of whatever the UART is driving accepts that character, it resets the UART by bringing the DTR or DataReady Reset line low.

For some simple UART uses, you instead want to just say "here is the data" and immediately get ready for more data. A bottom inverter does this, outputting a negative pulse every time a new character is ready to be received.

Thus, you use this inverter when you do not want or need handshaking. Note that the UART will hang after one character if you do not pull the IC's pin 18 low.

How are figures for this column drawn?

All four figures in this month's Hardware Hacker (and all those that have appeared in this column during the past few months) were "drawn" on an Apple IIe using the stock and popular ProDOS Appwriter word processor. The figure drawing process is sped up and helped along by some utility routines I have put together on my own. These figure textfiles are written in a special yet easy-to-use Postscript language. The Postscript text files are printed on a Laserwriter printer. The images are then photographically reduced to their final page size.

Note than the quality of these images is vastly better than anything that's currently available from any screen-oriented graphics routine on any microcomputer. This happens because an exact text description is far and away the most powerful, most flexible, and most exact way of wringing the ultimate performance out of the Laserwriter.

With the right utilities, exact text descriptions of electronic circuits can actually be done faster and much more compactly than by using screen graphics programs.

For instance, Fig. 4 took about 12 minutes to "draw." The final Postscript file ended up pretty much the same size and with just about the same complexity as an ordinary business letter.
Let's quickly sum up some of the key secrets of my new schematic drawing utilities:

First is a gray rubber grid that lets you stretch or squash the drawing to the size you need. Normally, one unit of the grid is defined as the space between adjacent pins on an IC package. You can turn the grid on and off as needed. Unlike screen graphics grids, you can easily use fractional grid values. No matter how fine the grid or how nervous the programmer, horizontal and vertical lines always stay that way.

Second is the use of opaque symbols. A symbol is something like a resistor, an IC or an inverter. Whenever these symbols are put down, they erase and then overwrite whatever happens to be under them. Thus, you could put down one continuous wire first, and then drop a resistor on top of it. This is far faster and easier than worrying about exactly where each connection to each part of each symbol has to go. Symbols can later be slid along a wire for best final appearance.

Third is the use of action points. Rather than define the position of a symbol from its corners, you define it from a logical connection point instead. The action point of a resistor would be its center. The action point of an IC pictorial is the center of pin one, and so on.

Fourth is the use of an automated DIP pictorial generator. To draw a complex 40-pin integrated circuit, you use a special routine. All you have to tell this routine are the number of pins, the name of the chip, and two strings of callouts as needed for the top and bottom pin rows. Thus, an entire DIP pictorial is drawn with a few dozen keystrokes! Complement bars are easily added when and as needed with a companion routine.

Fifth is the use of slide-under wiring. Since the symbols are all opaque, they will position themselves on top of any wires and will completely erase any wire under them. By putting the wire descriptions early in the textfile and the symbols late in the textfile, all of the wires will magically stop at the exact edge of each symbol. Usually, you put your symbols down first, and then slide the wires under them. This mind-boggling trick is done simply by entering wire descriptions above the symbol callouts in your textfile.

Sixth is the use of predefined symbols. To show a resistance omega, you just type the word "ohms," and so on. Things like arrowheads, clock pulses, and connection dots are similarly handled. You do not have to define these in the textfile for a particular schematic; they are already predefined in the preloaded schematic utility routines.

Seventh, and finally, is the use of self-breaking wires. Note the white space in Figs. 3 and 4 every time a horizontal wire ducks under a vertical one. This gets handled automatically by drawing a fat white vertical wire and then redrawing a thin black vertical wire on top of the white one. Special adjustments are made to keep from erasing portions of the wire endpoints. You can pick no breaks, vertical breaks, or horizontal breaks per your drawing style.

These same ideas will work very well for quick and simple printed circuit layouts. More on this some other time.

I'd be most happy to send you some free listings of these utility routines, along with a few mind-boggling demos.

Just call or write.

---

**NAMEs AND NUMBERS**

<table>
<thead>
<tr>
<th>Adobe Postscript</th>
<th>Digi-Key</th>
<th>OKI Semiconductor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1870 Embarcadero S120</td>
<td>701 Brooks Ave. South</td>
<td>650 North Mary Avenue</td>
</tr>
<tr>
<td>Palo Alto, Ca 94303</td>
<td>Thief River Falls, MN 56701</td>
<td>Sunnyvale, CA 94086</td>
</tr>
<tr>
<td>(415) 852-0271</td>
<td>(800) 344-4539</td>
<td>(408) 720-1900</td>
</tr>
<tr>
<td>Apple Computer</td>
<td>Mitsubishi</td>
<td>Rockwell International</td>
</tr>
<tr>
<td>20525 Mariami Avenue</td>
<td>1050 East Arques Avenue</td>
<td>3310 Miraloma Avenue</td>
</tr>
<tr>
<td>Cupertino, CA 95014</td>
<td>Sunnyvale, CA 94086</td>
<td>Anaheim, CA 92803</td>
</tr>
<tr>
<td>(408) 966-1010</td>
<td>(408) 730-5900</td>
<td>(800) 854-8099</td>
</tr>
<tr>
<td>Intersil</td>
<td>NEC Electronics</td>
<td>SAMS</td>
</tr>
<tr>
<td>10710 N. Tantau Avenue</td>
<td>401 Ellis Street</td>
<td>4300 West 62nd Street</td>
</tr>
<tr>
<td>Cupertino, CA 95014</td>
<td>Mountain View, CA 94039</td>
<td>Indianapolis, IN 46206</td>
</tr>
<tr>
<td>(408) 996-5000</td>
<td>(800) 632-3531</td>
<td>(800) 428-SAMS</td>
</tr>
</tbody>
</table>