Memory for the Galaksija "Galaksija 48K"

Translated from Issue 2 of Računari u vašoj kući (page 88) by Vlado Vince

For a computer to function as a complete unit, all its components need to work well with each other. You can't make use of a powerful operating system with a small amount of RAM (Sinclair QL, apparently, doesn't have enough even with 128K!), and copious amounts of memory won't get used well with an inadequate OS. What amount of memory is, then, appropriate for the Galaksija computer? Skeptics would say that it already has enough, but any Galaksija owner who tried to write a semi-serious BASIC program knows that this isn't true. Those who say that the Galaksija with 54K of memory will feel "like it's wearing somebody else's suit, three sizes too big" are probably closer to the truth. Loading programs into larger memory, we agree, can take a whole eternity. It's not all about loading, though, even if there's no way to improve it.

Large memory may seem awkward on the Galaksija computer, but it may compensate for some of the lower specs on this computer. Domestic computer fans, who, due to circumstances, have to get low powered machines below their requirements, often look to OS improvements as their only option. Galaksija's OS is perfectly open to such modifications -- Galaksija just needs some extra memory. Larger memory, therefore, is a prerequisite for further hardware expansions, such as fine graphics, and for more complex programs. Who will write serious programs when all you've got is 6K?

Even though current needs can't be a reliable measurement for determining the memory requirements --those will be determined by the hardware and software development which no-one can predict right now-- every Galaksija owner will have to figure out their own requirements. With the already built in memory, 16K presents the most reasonable size. If that is still not enough, memory can then be expanded by an additional 16K or 32K by adding just 2 or 4 more ICs. Don't be afraid of taking it one step at the time -- if you start with smaller memory and need to upgrade later, you will likely find cheaper parts in the future.

Image description: Galaksija 48K: memory expansion increases Galaksija's memory based on the number of ICs: by 16K, 32K or 48K -- thanks to the flexible address decoder, the expansion can be adjusted in the memory map in increments of 4K

Static or dynamic?

While there was some debate about what type of memory to use during the initial design of the Galaksija, there's no question anymore. The static memory used in the Galaksija is so much more expensive now that we would only recommend it in the most extreme circumstances. Currently, a KB of dynamic RAM is 8 times cheaper than static, meaning that for 6KB of static RAM in the original Galaksija you can get 48K of dynamic RAM! Static RAM was chosen for the Galaksija in order to keep the board design simple

enough for DIY builders. So what is the actual difference between dynamic and static RAM?

The memory cell in static memory is based on a flip-flop -- a fairly complex electronic circuit which is achieved by using five to six transistors. Once written, the content of this cell remains stored until it is overwritten, or the computer is turned off. All the circuits necessary for this memory to function are built into the IC itself, so the procedure of writing and reading data is completely spontaneous. Static memory is, however, quite bulky - any significant memory configuration would require many ICs and a huge PCB. Manufactures have so far managed to put 8192x8 cells into one chip (8KB) but such a chip will run you $50!

With dynamic memory, the cell is built around a capacitor. Since they take way less space than cells built around transistors, capacitors allow for a much more dense packaging, and a smaller price per byte. Micrometer dimensions, however, include a "micrometer" capacity, so the cell is emptied very fast and its contents are lost. The best dynamic memory can save data for 4 miliseconds tops. That's why its contents have to be refreshed constantly. Refreshing dynamic memory is a trivial process for the Z80 CPU, which can perform it after every machine cycle.

The second specificity of dynamic memory, however, isn't so simple and it requires some extra hardware and a specific procedure for writing and reading data. Memory cells in dynamic memory are organized in rows and columns, but direct access is only possible for rows. That's why the address is reported to the memory twice -- like with 8 bit processors. First you write the rows and then --during the same machine cycle-- the columns. The cell will only then be available to receive or send its contents. This procedure and some extra hardware are the complications when working with dynamic memory. It's fairly trivial, and mostly insignificant when accounting for the price, and it doesn't affect reliability. Dynamic memory is literally used in millions of home computers -- there are no reasons why we shouldn't use it in the Galaksija.

Text above the schematic: Memory is by far one of the most essential elements that determine a computer's capabilities. Regardles of power and capacity of its operating system, contemporary computers use all the memory space that the CPU allows. Such a concept, understandably, wasn't possible with the Galaksija computer, but such an expansion becomes mandatory for all who wish to do more serious work with our machine. Even though these are difficult times for hobbyists due to the state of the global chip market, we've developed a memory expansion for the Galaksija which brings Galaksija closer to other "serious" computers. The expansion allows the increase of memory in blocks of 16K and it can be connected to either the 2K or 6K Galaksija.

*description below the components list: The mounting schematic for the Galaksija 48 expansion: Even though we've used a double sided PCB, there are a few jumpers. The jumpers that go underneath memory chips are used to shorten the connection between capacitors and the ground, which increases their efficacy in stabilizing the power input for dynamic chips. Jumpers beneath and above the 74LS159 chip are used to program the address decoder. The picture shows the setup for the 6K Galaksija (with all 3 static

memory chips)

Both everything and something

There are four types of dynamic memory on the market. The first one, 4116, 16384 bits x 1, is the cheapest, and perhaps the most logical choice. 8 chips, necessary for 16K will run you about 2000 dinars. These chips, however, require triple power supply, and larger capacities require a complicated board and many ICs. Its configuration and technical design is outdated. The double sized memory used in the ZX Spectrum, using 32768 x 1, is a much better choice. This type of memory --it took a long time to find this out-- isn't manufactured anywhere. It's actually 4116 memory (65536 x 1) that has one invalid section!

That means there's really only two types remaining. 8 chips of 4164 memory, costing 40 pounds, will get us 64K! Even though we can't use all of it in the Galaksija or another computer (we need to leave some space for the ROM, even though RAM can work "underneath" the ROM, under some circumstances), this configuration is the most elegant solution. The memory covers all the areas in the memory map, including the ROM 2 space, and it can take the place of one or two static RAM slots (D and E). Those Galaksija owners who haven't gotten those chips yet can get half their memory expansion for the same price.

For those who love complete solutions --and complete solutions are always more elegant and cheaper than incomplete ones-- this would be the only choice. We developed a prototype of this solution in our newsroom and it showed great results, and we will use it to develop further system software. However, we decided to go with another variant. Why? This solution has one serious "philosophical-financial" issue -- it leaves Galaksija owners with an all or nothing choice. We're afraid that most fans would be forced to chose nothing instead of everything. It would be a beautiful, but expensive and inaccessible solution.

The fourth type of memory, 4416 which is organized in 16384 x 4 bits, maybe isn't as flexible in filling out the memory map, but it includes the best of both worlds. You can fully expand the memory, but you don't have to do it at the same time -- you can do three separate expansions of 16K. One such block, consisting of 2 x 4416 chips, is cheaper than a single 6116! This type of memory is relatively new and it's a bit harder to find, as it's currently only available in England, but not for long.

Memory chip 4416 has 8 address lines - A0-A7, and four data lines D0-D3, which means that one IC can only remember half a byte (4 bits). That's why every half a byte uses one chip. Control inputs WE (write/read), RAS (writing row address and refresh), CAS (writing column address and chip select) and G (output permission) are used to control the procedure of writing and reading data. The chip can work in a few different modes and with rather complex memory schematics. We chose, naturally, the simplest one.

"Dynamic" procedure

The electronic schematics for the Galaksija 48k is a classic one for this type of circuit, without any extravagant solutions or complicated timing circuits. Row and column addresses are provided through the two 74LS157 multiplexers. Depending on the state of the control input S, multiplexer sets either the address received from the A input (A8-A13) or the B input (A0-A7) to the Y outputs. At the start of the writing or reading cycle, the multiplexer sets the row address on the inputs A0-A7, and then about a hundred nanoseconds later it sets the address for columns to inputs A1-A6. The row address is written into the memory chip on the falling edge of the RAS (row address select) signal, while the column address is written on the falling edge of the CAS (column address select) signal. About a hundred nanoseconds after completing the address the output buffers are activated and the data can be read.

The procedure of writing the addresses is controled by the MREQ signal through a few OR gates with appropriate delay circuits. The address inputs of the memory chip always contain row addresses and they are caled every time that the CPU talks to the memory via the MREQ signal -- regardless of whether it wants to read, write or refresh memory cells. If it wants to write or read, WR signal will get the memory chip ready, and RD or WR will open the OR gate N6. Since it will have already written the row address, MREQ will "reconnect" the multiplexer (after a certaiin delay (R1, C1, N6) and bring it to A1-A6 column address. After this address is stabilized, the output of the delay circuit around N7 (R2, C2) will see the CAS signal and write it into the memory chip. While writing the data, the chip will get the column address along with the data (the so called early write modality -- writing with the CAS signal). While reading, the data will appear about 100 nanoseconds after writing the column address.

The CAS signal, meanwhile, controls the input/output buffers, the data cannot be written nor read unless the input which it controls isn't low. That means that it can be used to select a certain memory block. Address decoder 74LS159 determines the memory block that this signal will be directed to. The decoder covers the whole memory area in 4K blocks. Since it's a model with the so called open collectors, the outputs can be connected in parallel without any limitations. For a block of 16K you have to connect 4 outputs. The Galaksija 48K schematic allows for two options -- connecting to the 2K of existing memory (connect points A-B, C-D, E-F) and 6K (B-C, D-E, F-G). What should those who built the 4K Galaksija do? You should either buy another 6116 memory chip, or cut the CS line going towards the "D" memory. In the first case you can expand the memory to 64K, while in the second you can only get to 60K.

It may seem that using a decoder is too much for such an application. However, this was the only way to build a flexible sistem. Even those who have no intention of ever fully expanding the memory will have use from it -- we've exposed the CS signals for the last 12, or 16K on the connector. They can be used for some other use without a need for another connector. For example, they can be used to expand the ROM in combinations of 1x16K, 2x8K or 4x4K! Those who plan not to use the last 16K of RAM should cut the connections between legs 14 and 15 and 15 and 16 to separate CS signals in the last 16K of address space.

The PCB

The PCB is double sided due to its rather complex structure. We know that this will make some hobbyists unhappy. This circuit requires a professional board, however. For those interested and ready to wait, we'll organize the manufacturing and delivery of the PCB. For those who can't wait, we're publishing a 1:1 drawing. Those who decide to build the board themselves will have to get special pins for the ICs, like those on our photos, or to solder the chips directly. This is of course only necessary for the DIY boards. On the official boards the holes will connect on both sides and you won't have to worry about making contact. This type of board, of course, only requires soldering on one side.

... and the connector

Galaksija's connector, for some inexplicable reason, doesn't have the 5V and RD signals exposed, which are necessary for the memory expansion to work. That's why the connector on the Galaksija has to have some modifcations before the expansion is connected. 5V should be brought to pin 1 on the connector with a piece of wire, while RD should be brought directly from pin 21 of the Z80 to pin 22 on the connector. Pin 22 is normally ground, so this connection has to be cut. In order to open up space for the CS signals, all connections between pins 6 and 15 need to be cut too. Another connector is located on the rear of the memory expansion. It will allow for additional peripherals to connect at the same time. That could be an eprom programmer, a tone generator, a printer interface or something else.

Memory bugs

It's easy to test the memory expansion. Galaksija actually performs this test during each initialization -- the result is immediately visible. Moreover, you could just type PRINT MEM and you will see the number which will, depending on the amount of memory you put in, make your head spin? Unfortunately, that's not the case. As weird as it may sound, Galaksija's OS does not provide full support for the memory expansion. It includes a few bugs that make working with external memory difficult in some specific situations. The most serious bug happens with the full memory format. Instead of treating the free 54K of memory as it would any other amount, Galaksija writes 0 in the "end of memory" variable and acts like there is no free memory and it stops working. How is that possible? During the memory test, Galaksija puts the "end of memory" variable to the first location after the last one. And the first location after FFFF is 0000!

This will only trouble those who build a 48K expansion on top of the existing 6K. The solution is simple, but inellegant, and it's rather cumbersome. You need to write FFF0 into the "end of memory" variable (&2A6A). This will allow the Galaksija to work. That is, until it's turned off. Manually setting the "ramtop" is unfortunatley the only solution in this situation. All the other combinations work well, but that means that we would need to give up a whole memory block, which is in our case 4K, to get around not needing this fix.

To make matters worse, the PRINT MEM command starts to struggle as we increase memory. It works well until we get to 22K of RAM (until address 32767). Afterwards it shows a negative number that has no relation to the actual memory size. With 16K PRINT MEM shows 21445, from 32K -27706, and with 48K -11322! Checking the "end of memory" variable by typing PRINT WORD (&2A6A) also shows negative numbers -32768 (16K), -16384 (32K) and 0 (48K). In the space above 32K of RAM Galaksija only accepts hexadecimal numbers for the BYTE and WORD commands (instead of also decimal numbers). All these limitations, however, have no impact on the actual working with memory - it normally accepts both BASIC and machine programs.

*descriptions next to the drawings on the last page: Connector and the connector board. The sandwich system of mounting allows connecting multiple expansions along the memory expansion, such as an EPROM programmer or a sound generator. The pins on the expansion connector have been modified with new signals (RD, CS1, CS2, CS3 and +5V)

Connector board on the Galaksija: in order to allow for easier connection of peripheral devices the board has to be modified. All the modifications should be done on the top side of the board. You should use a precision knife, a soldering iron and two pieces of wire.*

Project and text: Jova Regasek