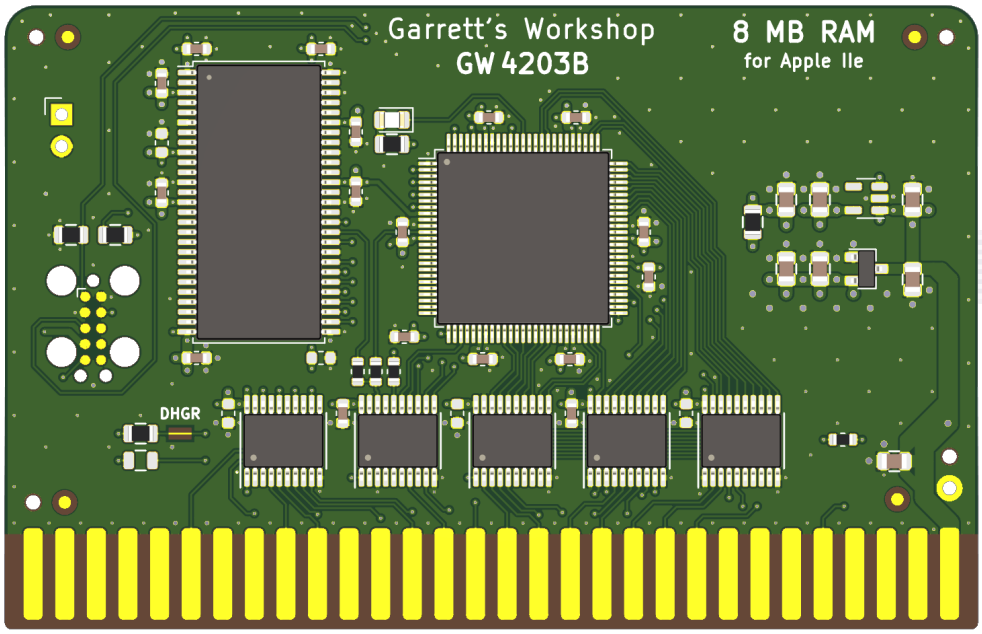
Garrett’s Workshop

GW4203B “RAM2E II”

RAM Expansion Card for Apple IIe

Developer Note



RAM2E II was designed by Zane Kaminski and Garrett Fellers

Theory of Operation

The operation of RAM2E II is somewhat different from that of other expansion RAM cards for the Apple IIe. While other cards are implemented with asynchronous DRAM chips, RAM2E II uses modern synchronous DRAM (SDRAM).

Since the IIe's memory expansion slot was designed for use with asynchronous DRAM, additional circuitry on the RAM2E II card is required to interface the IIe with SDRAM. There are numerous differences between asynchronous DRAM and SDRAM, but the most significant is that in SDRAM, operation is pipelined over multiple clock cycles. Although SDRAM is a much newer technology than asynchronous DRAM, and therefore much faster, multiple clock cycle "steps" are required to perform an SDRAM access.

RAM2E II's logic circuitry is implemented in a single CPLD which runs from the Apple IIe's C14M 14.31818 MHz master clock signal. A buffered copy of this clock is supplied to both the RAM2E II's CPLD and its SDRAM. Running from the Apple's master clock, the RAM2E II translates auxiliary memory and 80-column video access commands issued by the Apple II into SDRAM commands which implement the same functionality.

The RAM2E II state machine runs from the C14M master clock of the Apple IIe. A 4-bit state counter is reset to 0x1 at the beginning of each PHI1 period and counts to 0xE in a 14-clock cycle. In the last two clocks of a 16-clock "long cycle," the state counter is equals 0xF. State 0x0 is only used during initialization.

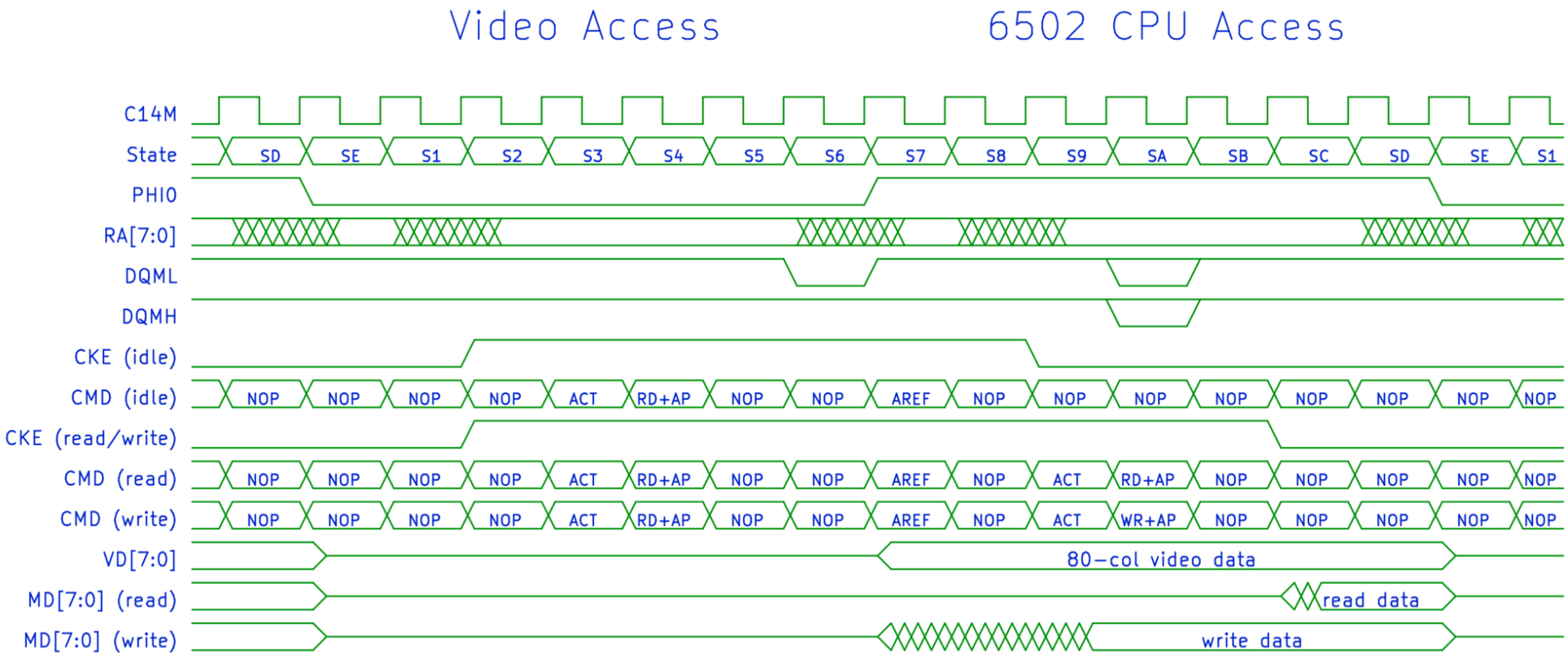
To implement the auxiliary memory card functionality, SDRAM read/write commands are issued based on the current state index, the /WE80 signal, and the /EN80 signal. Unlike a traditional auxiliary RAM card, the /RAS and Q3 signals are not used for SDRAM control at all. The SDRAM command and address signals are implemented as registered outputs.

The data bus, as input to the RAM2E II and output to the SDRAM, is implemented as an asynchronous tristate buffer, and the data bus output is implemented as a registered output latched at the falling edge in the middle of state 0xC. Similarly, the video data bus output is registered at the falling edge of state 0x6. Both the video and 6502 data buses are output using 74AHCT-series buffers running at 5V. 74AHCT was chosen for its low power consumption, fast propagation delay, and slow output edge rate. Moreover, the 74AHCT-series outputs are desirable for their 3.8 V Voh specification at 8 mA of source current and 4.5V Vcc. This allows RAM2E II to meet the Vih specification of newer 65C02 processors with "pure CMOS" input buffers.

RAM2E II also supports a proprietary command set which allows software to adjust the RAM capacity and access other features. The command set is divided into "supported commands" and "nonstandard commands." Supported commands are considered stable and safe to use by developers. All stable commands will be supported in all future versions of the RAM2E. Nonstandard commands are for internal use by Garrett's Workshop and are not recommended for use by third-party developers.

Timing Diagram

The timing diagram given below shows the behavior of the major signals in the system, including the SDRAM command sequence used when reading and writing auxiliary RAM.



Information for Developers: RAM2E II Command Set

RAM2E II supports a proprietary command set to facilitate software access to the special features of the card (for example to adjust the RAM capacity). The content of a command consists of an 8-bit command number and an 8-bit argument. Commands are issued by repeated writes to the RAMWorks bank register at $C073. To issue a command, a series of eight write accesses to the RAMWorks bank register at address $C073 is required. Each of the eight bytes must be written within seven PHI0 clocks of the submission of the previous byte or the command will not be accepted. This prevents ordinary RAMWorks bank switches from inadvertently triggering a command submission. In the first part of the command sequence, six magic numbers must be written in sequence to $C073. The magic number sequence which must be written is $FF, $00, $55, $AA, $C1, $AD. Following writing the magic number sequence to $C073, the command number and then argument are written to $C073. Once all eight bytes are submitted, the command is executed. If eight or more PHI0 cycles elapse between submission of subsequent bytes of the command sequence, or if an incorrect magic number is submitted, the command sequence detector resets and the command sequence must begin again for any command to be accepted. After a command is submitted, unless otherwise specified in the command description, the RAMWorks bank address will be equal to the argument submitted, as the argument is the last byte written to $C073 as part of the command sequence. For more information on this command set, see the RAM2E and GWRAM utility program source code on the Garrett's Workshop GitHub page at http://github.com/garrettsworkshop.

Sample code in 6502 assembly language is given below to submit a command to the RAM2E II. The code assumes that the command number is stored in the X register and the argument is stored in the Y register.

```
; Routine to submit a RAM2E II command
; Assumes command is in X register and argument is in Y register
; First reset command sequence just in case
lda #$00
sta $C073
sta $C073
; Send magic numbers to start command (FF 00 55 AA C1 AD)
lda #$FF
sta $C073
lda #$00
sta $C073
lda #$55
sta $C073
lda #$AA
sta $C073
lda #$C1
sta $C073
lda #$AD
sta $C073
; Send command and then argument (in X and Y registers respectively)
stx $C073
sty $C073
```

Supported Commands

Supported commands are meant for use by third-party application developers. A table of supported commands is given below:

Command Name	Num.	Description
NOP	$00	No special meaning: RAMWorks bank is set to argument after command is executed.
SetRWBankFFMAX **SetRWBankFFSPI** **SetRWBankFFMXO2** **SetRWBankFFLED**	$FF $FE $FD $F0	This family of commands sets the RAMWorks bank to $FF after command is executed, independent of argument value. Other auxiliary RAM cards will set their bank to the argument provided, rather than $FF. This allows software to detect the presence of a RAM2E card. Not all RAM2E II cards respond to each message. Original RAM2E II cards without LED indicator respond only to the SetRWBankFFMAX command. Subsequent models with LED respond to the SetRWBankFFLED command and one of the three other commands.
RWMaskSet	$E0	Sets the AND mask applied to the RAMWorks bank, thus setting the RAMWorks capacity. For example, a mask argument of $00 corresponds to a capacity of 64 kB, whereas $7F corresponds to a capacity of 8 MB.
LEDSet	$E2	Enables or disables the activity LED. To enable the LED, set bit 0 of the argument to 1. To disable the LED, set bit 0 to 0. Bits 7 through 1 of the argument must be 0. This command is only supported on RAM2E II cards with onboard activity LED.
LEDGet	$E3	This command is used to check whether the LED is enabled or not. It works similarly to the SetRWBankFF family of commands. If the LED is enabled, the current RAMWorks bank is set to $FF after the command is executed. Otherwise, the RAMWorks bank register is set to the argument value. This command is only supported on RAM2E II cards with onboard activity LED.

Nonstandard Commands

In addition to the canonical "supported commands," there are "nonstandard commands," comparable to "unpublished APIs." These commands are implementation-specific. No individual RAM2E card supports them all. Repeated use of the CFGReset command over tens or hundreds of power cycles may significantly decrease the remaining lifespan of the nonvolatile memory used for user setting storage. We list these commands here for completeness but do not recommend that third-party developers attempt to submit nonstandard commands to a RAM2E card.

Command Name	Num.	Description
BitbangMAX	$EA	Shifts configuration bit into holding register in preparation to write to CFG flash. Bits 0 through 5 of the argument must be 0. Bit 6 of the argument is the configuration bit to shift in. Bit 7 of the argument controls clocking of the shift register and must be 1.
BitbangSPI	$EB	Allows for output-only SPI bit-bang control. Bit 0 is MOSI. Bit 1 is SCK. Bit 2 is CS. Bits 7 through 3 of the argument must be 0.
BitbangMXO2	$EC	Argument byte is shifted into the wishbone address register, then the data register.
RWMXO2	$ED	Submits a read/write on the wishbone bus. Bit 0 is WE. Bits 7 through 1 of the argument must be 0.
ResetMAX	$EE	Completely erases the RAM2E II card's CFG flash. This command is intended to repair the CFG flash if it is put in an invalid state. Avoid frequent use of the ResetMAX command, as it causes wear-leveling information in the CFG flash to be erased, thus significantly decreasing the CFG flash's remaining lifetime. This command can only be executed once per power cycle. Subsequent invocations of the ResetMAX command are equivalent to a NOP command, preventing an errant program from writing to flash repeatedly and needlessly reducing the lifetime of the CFG flash. Argument must equal $00. Future versions of RAM2E II may not support the ResetMAX command.
PrgmMAX	$EF	Writes configuration data loaded using the BitbangMAX command to CFG flash. This command can only be executed once per power cycle. Subsequent invocations of the PrgmMAX command are equivalent to a NOP, preventing an errant program from writing to flash repeatedly and needlessly reducing the lifetime of the CFG flash. Argument must equal $00. Future versions of RAM2E II may not support the PrgmMAX command.

All other command numbers are reserved and should not be submitted to the RAM2E II card.