GL85 Microprocessor Structural Hierarchy

This circuit is similar to the 8085 microprocessor. One major difference is the use of separate 8-bit input and output busses in place of the 8-bit bidirectional $AD_{0\text{-}7}$ ADDRESS/DATA bus. It must also be pointed out that no attempt has been made to verify accuracy of signal timing at the I/O pins relative to that of any real implementation of the 8085 microprocessor.

The top level structural model is named GL85. It contains eight structural submodels, as shown in the hierarchical tree. Some of the first level submodels contained in GL85 contain only submodels, and perhaps some very simple glue logic. For example, CTL_LGC2 contains the 9-state state machine *Stater,* the 5-state state machine *M5*, and the two submacros *Hltlogic* and *Hldlogic* which contain the Halt and Hold flip-flops.

Overview of the GL85 Microprocessor Model

The GL85 instruction set, like its 8085 counterpart, is grouped into five categories, these include:

data transfer
arithmetic
logic
branch
stack, I/O, and machine control.

The instructions each have one, two or three bytes. The first byte is always the op-code. The eight-bits of the op-code are numbered I_7 to I_0, where I_7 is the high-order bit and I_0 is the low-order bit. The op-codes are broken into three fields. The high-order two bits of the op-code, I_7 and I_6, select one of four quadrants of the op-code map. The next three op code bits, referred to as DDD, are often used to represent one of eight destination registers, and, likewise the low order three bits of the op-code, referred to as SSS, represent one of eight source registers for many of the instructions.

Operation of the GL85 proceeds under control of two state machines, referred to here as the M-cycle and the t-state state machines. When the op-code byte is read in and decoded, it will require anywhere from one to five M-cycles, and each of the M-cycles requires from three to five t-cycles. The first t-state of the first M-cycle, denoted T1, always causes the contents of the PC (program counter) to be output to the address bus. In the second t-state of M1, denoted T2, the PC is incremented. Then, in T3, the op-code is clocked into the IR (instruction register).

Consider the Move Register-to-Register instruction:

MOV r1,r2

This is a single byte instruction. It requires one M-cycle that is sub-divided into four t-cycles. This instruction causes the contents of register r2 to be placed into register r1, where r1 and r2 can be any 8-bit register from the set {A,B,C,D,E,H,L}. The three bit SSS and DDD fields are decoded as follows:

DDD/SSS	**Register**
000	B
001	C
010	D
011	E
100	H
101	L
110	(memory)
111	A (accumulator)

The first three t-states, as previously indicated, cause the PC to be put onto the address bus, then the contents of PC are incremented, and the op-code is latched into the IR. On state T4 the data transfer takes place between the source and destination registers.

Data transfers can also occur in 16-bit groups. Consider the '*load register pair immediate'* instruction:

LXI rp,data

This is a three byte instruction. The first byte is the op-code. Bits I_7 and I_6 of the op-code contain the value 0,0. Bits I_5 and I_4 designate a register pair, according to the following decoding:

RP	**Register Pair**
00	B-C
01	D-E
10	H-L
11	SP (stack pointer)

The remaining four bits I_3 to I_0 of the op-code contain the value {0,0,0,1}. The second and third bytes of the instruction contain the data that is to be moved from memory into the register pair. Byte 2 of the instruction goes into the low order register of the register pair and byte 3 goes into the high order register. So, if the first byte contains the hex value X'01, then the value contained in the second byte of the instruction goes into register C and the byte contained in byte 3 goes into register B.

The LXI instruction requires three M-cycles. The first M-cycle has four t-states, where state

T4 is used for internal instruction decoding. The second and third M-cycles each has three t-states. T1 of the second M-cycle causes PC to be put out on the address bus. T2 causes the PC to be incremented and during T3 the second byte of the instruction is loaded in the low order register of the register pair. The third M-cycle is identical in behavior except that byte 3 is loaded into the high order register of the register pair.

The Structural Model

In examining the modules that make up the GL85, note that there are a total of 50 structural modules and 34 of them are listed on the hierarchical map. The remaining 16 are common utility modules. They are listed here, followed by a description of their function:

buf8	- 8 buffers
inv8	- 8 inverters
decod2_4	- 2-to-4 decoder
decod3_8	- 3-to-8 decoder
g4bctr	- 4 bit counter
mux2to1	- 2-to-1 multiplexer
mux_4bit	- 4-to-1 multiplexer
ocnand	- 8 NAND gates used as open-collector
parity	- 8-bit parity generator
prioenco	- 5-to-3 priority encoder
reg8bits	- 8-bit register (Data input multiplexed, Clock not gated)
reg_8bit	- 8-bit register (Clock gated)
regpair	- two 8-bit registers (Uses reg_8bit)
regpairs	- two 8-bit registers (Uses reg8bits)
sn54181	- 4-bit ALU
sn85150	- 10-to-1 selector

INST_REG

The module INST_REG contains an eight-bit register that is used to hold the op-code currently being decoded. It also contains three decoders. Two three-to-eight decoders are used to decode the SSS and DDD fields of the op-code. The SSS field is decoded into signals ID0 to ID7. The DDD field is decoded into signals ID8 to ID15. A two-to-4 decoder decodes the two high-order bits into signals ID16 to ID19. These twenty signals ID0 to ID19 are then used to control signal transfers within the microprocessor model, in conjunction with the states defined by the M-cycle and the t-state state machines.

These signals are active when low. Consider the LXI instruction above, where the op-code is X'01 and the three fields are 00 000 001. Signal ID1 is low, ID8 is low, and ID16 is low. By using these 20 signals, as well as the control signals emanating from the M-cycle and the t-state state machines, it is possible at any time to identify exactly what state the microprocessor is in, and what op-code is currently being decoded in the IR.

DATAADDR

This module contains the MAR (Memory Address Register) and the MDR (Memory Data Register). It includes two copies of the REG_8BIT register macro, as well as the OCNAND macro and four instances of the MUX_4BIT macro. MUX_4BIT selects either the 8- or 16-bit internal bus, or the 8-bit DATA input bus to be latched into the MDR. OCNAND is a collection of NAND gates that are used in an open collector mode to drive the 8-bit and 16-bit internal busses.

REGPAD

All of the 8-bit registers, except for the accumulator, are contained in the submodules that make up REGPAD. (The accumulator is in module OPRLOGIC). The location of the various 8-bit registers can be easily deduced from the submodule names. In addition to containing the H, L, D, and E registers, the submodule HL_DE_WZ contains two internal registers, W and Z, that are not visible to the programmer. The submodule G16BCTR contains a 16-bit counter that is used whenever incrementing the PC or SP (stack pointer).

REG_CTRL

When the op-codes and state machine states are decoded, signals are generated that cause the various registers to be loaded and/or their contents dumped out onto one of the busses to be transferred to some destination register. The various submodules contained in REG_CTRL decode ID0 to ID19, as well as the state machine states, in order to generate gating signals for these data transfers. PC_CNTRL and SP_CNTRL, as their names imply, generate gating signals used to load the PC and SP, respectively.

The modules REGCTRL0, REGCTRL1, REGCTRL2 and TEMPCTRL decode signals ID19 to ID0 together with the M-Cycle and Tstate states in order to generate signals that enable the various registers to be written onto and read from the 8-bit and 16-bit busses.

CTL_LGC1

This module generates the read, write, ALE, INTA, IOM, S1 and S0 signals. The component MCDCODE (machine cycle decoder) is used to generate signals, depending on the instruction type, for resetting the M-cycle state machine at the appropriate time.

MCDECODE also generates a signal if the instruction contained in IR has six T-states in the first machine cycle. It generates a 'onemc' signal if the instruction in IR has only one M-cycle and it decodes the instruction in the IR and generates a signal to reset the M-cycle state machine. It also generates internal signals wrinm2 (write in machine cycle 2), wrinm3 (write in machine cycle 3), and lastmc (last machine cycle). The module RDWRGEN generates the

signals RD, WR, ALE, INTA, S1, S0, and IOM.

CTL_LGC2

The modules HLTLOGIC and HLDLOGIC contain the halt and hold flip-flops, respectively. STATER contains the Tstate state machine and M5 contains the M-Cycle state machine. The clocking signal for the M-Cycle state machine comes from the Tstate state machine. The clocking signal is enabled by signals from MCDCODE.

INTERUPT

The interrupts are processed by four modules. VECTRGEN puts the interrupt vector onto the 16-bit bus during the INA cycle and RST instruction. INTRUPT1 consists of the MASK registers, the TRAP flip-flop, the RST7.5 flip-flop and the INTE flip-flop. It also contains the priority encoder as a component. INTRUPT2 recognizes a pending interupt in the last machine cycle of an instruction. It then generates INTA and INA signals. It also resets RST7.5 and TRAP flip-flops contained in module INTRUPT1. INTRUPT2 handles the serial input and serial output functions and stores the INTE flip-flop value before the occurrence of a TRAP.

ALULOGIC

This module handles all ALU related operations. The 8-bit ALU is contained in module ALU8BIT. FLAGUNIT contains the flag register and the flag control logic. The flag register holds the flag values that result from ALU operations, including generation of sign, parity, zero detect, auxiliary carry and carry bits. Three of its eight bits are unused.

SHFLOGIC generates a carry-out and shifted values in response to the RLC, RRC, RAL and RAR instructions. ALU_CTRL generates function select codes for the SN54181 ALU by decoding the op-code in the IR and determining what, if any, ALU function must be performed. It also generates the M signal for the SN54181 and it generates an enable for a carry-in to the SN54181. OPRLOGIC contains the TEMP and the Auxiliary ACC register. It also contains a multiplexer that selects the source for the Auxiliary ACC connected to the A-port of the ALU. Another 8-bit multiplexer selects either the TEMP register or the Carry and Auxiliary Carry as the source for the ALU B-port.

General Data Flow

We will describe the architectural features of this model of the GL85 microprocessor through the use of schematics that tie together the registers and the control logic found in the various submodules. This discussion will be at an architectural level and will only include enough detail to illustrate the principles of operation of the model. From this description the reader can fill in the remaining details.

The 8-Bit Bus

We will use the bus figures to describe data flow through the microprocessor. The GL85 model has two busses, an 8-bit bus and a 16-bit bus. There are 15 8-bit sources for the 8-bit bus. Each of these sources is controlled by a gating signal and is connected to the bus by means of a module called OCNAND, which is simply a collection of eight NAND gates. For example, in the figure of the 8-bit bus, we see that the 16-bit stack pointer register is broken into two 8-bit parts. The low order eight bits are gated by a signal called SP0OUT which connects to one of the inputs of each of the eight NAND gates in OCNAND. Each of the eight bits of SP7:0 is connected to one of the eight NAND gates in OCNAND. If SP0OUT is low, then SP7:0 is blocked from the bus. If SP0OUT is high, then SP7:0 is enabled onto the bus. When SP0OUT is high, the microprocessor is designed such that all of the other control signals will be inactive (low). In effect, then, the bus is connected in an open collector arrangement with a pullup resistor.

Both the 8-bit and the 16-bit busses can drive a bus called DN, which is eight bits in width. This bus, in turn, serves as input to the registers. To understand the overall flow of information along this bus, consider the DAD instruction. This instruction adds a register pair (double add) to the contents of the HL register pair and places the result in the HL register pair. The op-code is 00rp1001. When adding the contents of the 16-bit stack pointer to HL, rp assumes the value 1,1, so the op-code becomes 00111001.

Recall, from previous discussion, that the op-code is broken up into three fields for decoding purposes. The op-code 00111001 then becomes 00-111-001. The high order two bits, when decoded, cause ID16 to assume the value 0 and ID17, ID18 and ID19 all assume the value 1. The low order three bits cause ID1 to assume the value 0 and the other remaining bits in the range ID0 through ID7 assume the value 1. The remaining field, instruction bits I_3 through I_5, are all 1's. Therefore, from the 8-bit bus figure, it can be seen that the NOR gate whose output is labelled DAD assumes the value 1. Also, the 2-to-4 Decoder causes RP(3) to assume the value 1, and RP(0) through RP(2) all assume the value 0.

From the detailed description of the individual instructions found in the Intel *MCS-80/85 Family*

User's Manual, it can be seen that (rl) is gated into ACT during the T1 state of the second machine cycle M2. In this case the symbol (rl) refers to the low order half, bits 7:0, of the stack pointer, while ACT refers to a temporary accumulator. To summarize, if the DAD instruction is being decoded, and if the register pair field has binary value 1,1, and if the microprocessor is in state (M,t) = (2,1), then the contents of the low order half of the stack pointer register are gated onto B8BO. The control logic shown on the left side of Figure 2 is contained in the submodule REGCTRL1. The stack pointer register is contained in the module BC_SP_PC.

That describes the activity for a single state of the microprocessor for the instruction DAD. For state (M,t) = (2,2) the register L (the low half of the HL pair) is gated into the TMP register and then the contents of ACT and TMP are added together in the ALU. In state (M,t) = (2,3) the result from the ALU operation is gated into the L register, and the carry is gated into the carry bit of the status register.

For machine state M = 3 the operations are similar except that the high half of the stack pointer ard the H register are added together, along with the carry bit from the previous operation, in order to generate the final sum which is deposited in HL. Again, a carry is gated into the carry bit of the status register. Other register pair combinations selected by I_5 and I_4 enable other registers onto the bus B8BO, and, of course, other instructions will, in a similar vein, enable the various registers onto the bus at appropriate times.

The 16-Bit Bus

The operation of the 16-bit bus will be explained with the aid of Figure 3.

(To be continued.)