Additional Control Memory Improves Performance of Low-Cost Microprocessors

Wolfgang Matthes
Franz-Mehring-Strasse 22
O-9006 Chemnitz
Germany

Introduction

In many "embedded systems" applications, the performance of low-cost microprocessors (e. g. 8-bit CPUs) is quite sufficient, with exception of a few time-critical operations, especially with respect to the control of real-world interfaces.
Occasionally it may be advantageous to extend the effects of some CPU instructions by means of simple additional hardware. The following design ideas will work well with all CPU types which have internally neither instruction nor data buffering. As example CPU, the popular Z 80 will be used.

Control Principle

According to Figure 1, additional control memory is addressed in parallel to the instruction memory during CPU instruction fetch. Its content will be loaded into a control register. Thus the width and therefore the effect of CPU instructions can be extended. Figure 1 illustrates three basic principles:

1. Circuitry outside the CPU/memory structure is controlled directly. The control words are essentially microinstructions, and the CPU acts like some kind of microprogram sequencer.

2. CPU address and/or data lines are used for special I/O purposes.

3. Information on CPU data bus is substituted by information from other sources, which allows for modification of the instruction sequence or for fast input operations, respectively.

An instruction fetch within a specific address range will cause the control register to be loaded and the mode of extended effects (called CONTROL MODE) to be set. In the example, a 4k-segment (FxxxH) of the 16-bit CPU address is the CONTROL MODE address range. Each instruction fetch outside this address range will cause CONTROL MODE to be reset and the control register to be cleared. A cleared control register should cause no extended effects. Instructions fetched within the CONTROL MODE address range will have no extended effects if the corresponding control memory locations contain zeroes. Thus instructions with and without extended effects can be intermixed freely, without any performance degradation which would otherwise be inevitable due to the necessary mode switching. Figure 2 shows details of the CONTROL MODE circuitry.

The width of the control memory could be chosen according to the particular application requirements, from one bit (if only one special effect is to be controlled) up to 40 bits or even more, if necessary. Control words are somewhat similar to microinstructions, thus the corresponding principles may be applied (horizontal vs. vertical formats, binary vs. 1-of-n encoding etc.), and the corresponding literature may be used as reference.

Conventional CPU access (within a second address range) to control memory is not required, but it may be provided to facilitate diagnostics and program loading/changing, respectively. Note: For dia-

gnostic purposes (especially for signature analysis) it is strongly recommended to provide something to suppress the extended effects (e. g. a DIL switch to disable the control circuitry).
Figure 3 shows an example of a control word format. Each control word is 16 bits in length and resembles somewhat to "horizontal" microinstructions. The following details are necessary to understand the particular operations and the corresponding hardware structures:

a) The example format has been chosen to demonstrate the basic principles (see Figure 1) all together. To that end, five types of extended machine instructions are provided:

1. NO EXTENDED OPERATION (not shown explicitely).

2. PUT IMMEDIATE: immediate values will be moved to outbound interfaces.

3. PUT REGISTER: contents of CPU registers will be moved to outbound interfaces.

4. GET REGISTER: information from inbound interfaces will be loaded into CPU registers.

5. SKIP: according to a selected external condition, an instruction will be executed or skipped (i. e. replaced by NOPs), respectively.

Such instructions could be made available to the programmer by means of appropriate assembler macro definitions.

b) The extended format is typical for MSI implementations where, with respect to package count, LSI/MSI logic is more expensive than memory. Consequently, a rather "horizontal" encoding has been chosen which requires minimum decoding effort.

The OPERATION field (bits 13,14,15) could be decoded by a simple 3-to-8 decoder (this kind of decoding has been assumed in the Figures 4 to 7).

Bit 12 (Disable Interrupt; DI) prevents NMI and INTERRUPT signals from being propagated to the CPU (see Figure 7). This bit is to be set if interrupts are to be disabled in a sequence of extended instructions.

Bit 11 (FORCE NOP) substitutes a NOP opcode (00H) on CPU data bus (see Figure 5).

The bits 8,9,10 (INTERFACE CONTROL) are used to encode various control activities directly.

The bits 0 to 7 constitute an EMIT byte which is used for various purposes (immediate, source/destination select etc.). It may be used to encode application-specific control activities, too.

In the following, the particular operations will be described according to the Figures 4 to 7:

Output of Immediates (PUT IMMEDIATE)

Parts of the control word (e. g. the EMIT field) could be directly delivered to interface lines. Other parts of the control word may contain a destination code, if necessary. The EMIT field of the control word could be supplemented by immediates of appropriate "Load Immediate" instructions (e. g. LD HL,nn). The width of the interface could be extended to a great number of bits. Example (Figure 4): 16

bits correspond to the immediates n,m in the instruction LD HL,nm and 24 bits correspond to the EMIT fields of the three control words accompanying the three bytes of the instruction, thus the good old Z 80 could deliver 40 bits in 11 machine cycles. It may be a drawback that the HL register will be loaded with the immediates n,m. This could be avoided by instruction substitution (three NOPs are forced on CPU data bus instead of the LD HL,nm instruction). Note: instruction substitution (see Figure 5) requires the FORCE NOP bit to be set in all control words. This allows to use even the CPU opcode as an immediate which would contribute an additional byte to the interface.

Extended Input (GET REGISTER)

Bit strings exceeding the CPU word size could be read by data substitution on CPU data bus. Many CPUs have load instructions which load an immediate twice as long as word length into a CPU register (e. g. Z 80 LD HL,nm). In CONTROL MODE, the fetch of the first byte of the immediate is accompanied by a first control word. This will cause the immediate to be removed from CPU data bus (by disabling the memory data bus driver) and the bus driver for the first byte of the inbound interface to be enabled. The fetch of the second byte of the immediate will have a similar effect. Thus the CPU will receive the 16-bit information from the inbound interface instead of the immediates (Figure 5). Notes:

1. The EMIT fields in the control words may be used to select a particular interface port.

2. The D-type flipflop in Figure 5 has been provided to prevent bus drivers from acting against each other during the decision interval.

Extended Output (PUT REGISTER)

In CONTROL MODE, the CPU address and data buses could be used together for output purposes. Thus the Z 80 could drive outbound interfaces with up to 24 data lines. To that end, a register-to-memory move instruction (e. g. LD (HL),A) is extended by a control word (see Figure 3). The corresponding hardware structure is shown in Figure 6. A PUT REGISTER extended instruction causes the memory access decoders to be inhibited during the write access (where the control word remains active). Thus the address and data buses are both available to transfer data to the selected interface port.

Conditioned Branches (SKIP)

The principle of substitution on the CPU data bus can be used to provide conditioned branches, subroutine calls, or skipping of arbitrary instructions, respectively, according to external conditions (Figure 7). A unconditioned branch, call, or other instruction to be skipped conditionally is accompanied by control words which select the particular condition. The control hardware decides whether to execute or to skip the instruction. In the latter case, the instruction will be removed from CPU data bus and replaced by a NOP (00H). The hardware must take the decision in the first cycle of the instruction and keep the result until the last cycle has been executed.

If necessary, wait states must be inserted during the decision interval. (Note: LOAD PULSE in Figure 7 is to be activated after the decision has been taken.) The hardware must suppress all interrupts during the substitution (the substituted NOP sequence may never be interrupted), or it must allow to save and restore the state of substitution.

Thus it is possible to inspect many external conditions directly (up to 256 or even more) which would otherwise require a considerably longer instruction sequence (output to select the condition -> input of the selected information -> bit test -> branch).

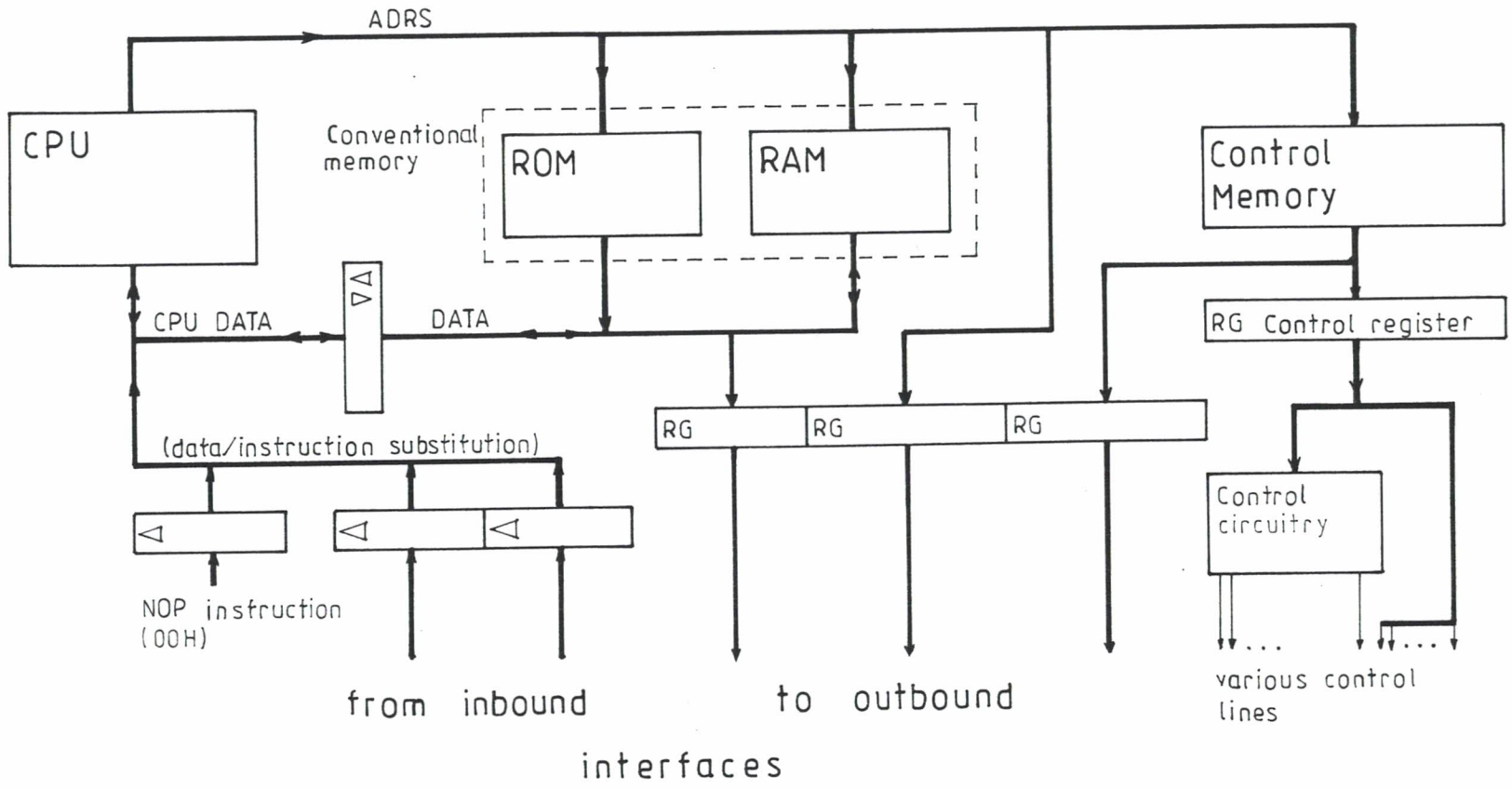

Figure 1: Microprocessor system with CPU instructions extended by additional control memory.

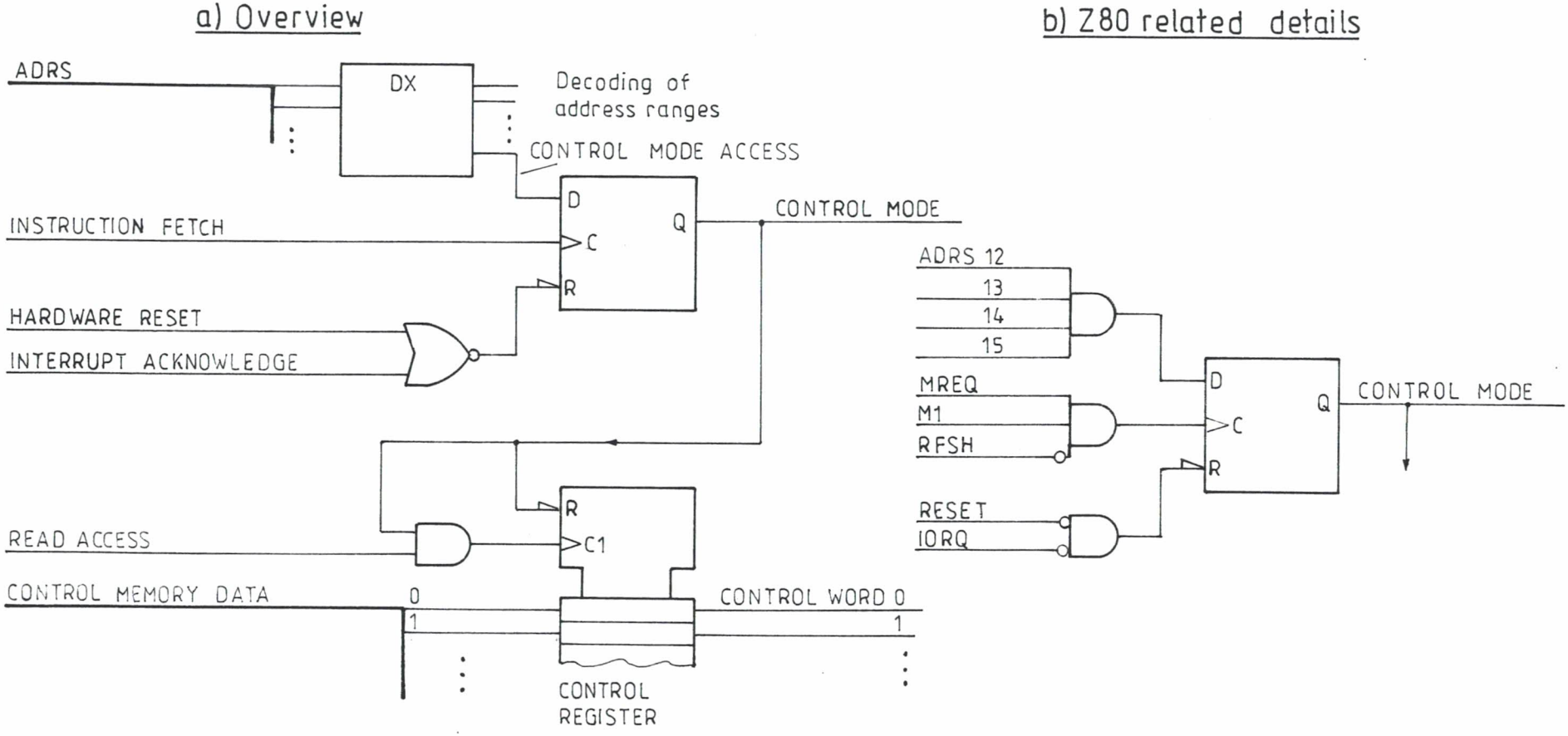

Figure 2: Basic CONTROL MODE circuitry.

Basic format:

Operations:

OPERATION (15 14 13)	DI (12)	FORCE NOP (11)	INTERFACE CONTROL (10 9 8)	EMIT (7 … 0)
PUT IMMEDIATE: 0 0 1	DI	FORCE NOP	LOAD 3, LOAD 2, LOAD 1	IMMEDIATE
PUT REGISTER: 0 1 0	DI	0	x, x, x	DESTINATION SELECT (optional)
GET REGISTER: 0 1 1	DI	0	x, ENABLE2, ENABLE1	SOURCE SELECT (optional)
SKIP: 1 0 0	DI	0	CPL, COND. LATCH, x	CONDITION SELECT

PUT IMMEDIATE

Cycle	Data bus	Control words
1	LD HL,nm OPC.	0 0 1 0 | 0 0 1 immediate 1
2	immediate m	0 0 1 0 | 0 1 0 immediate 2
3	immediate n	0 0 1 0 | 1 0 0 immediate 3

see text

PUT REGISTER

Cycle	Data bus	Control word
1	LD (HL),A OPC.	0 1 0 | 0 0 | x x x dest. sel.
2	content of A	above word is still effective

GET REGISTER

Cycle	Data bus	Control words
1	LD HL,nm OPC.	0 1 1 | 0 0 | x 0 0 x
2	1st byte	0 1 1 | 0 0 | x 0 1 sel. 1st src.
3	2nd byte	0 1 1 | 0 0 | x 1 0 sel. 2nd src.

SKIP

Cycle	Data bus	Control words
1	JMP OPC.	1 0 0 | 1 0 | CPL 1 x cond. sel.
2	ADRS (lo)	1 0 0 | 1 0 | x 0 x x
3	ADRS (hi)	1 0 0 | 0 0 | x 0 x x

Figure 3: Example control word formats and operations.

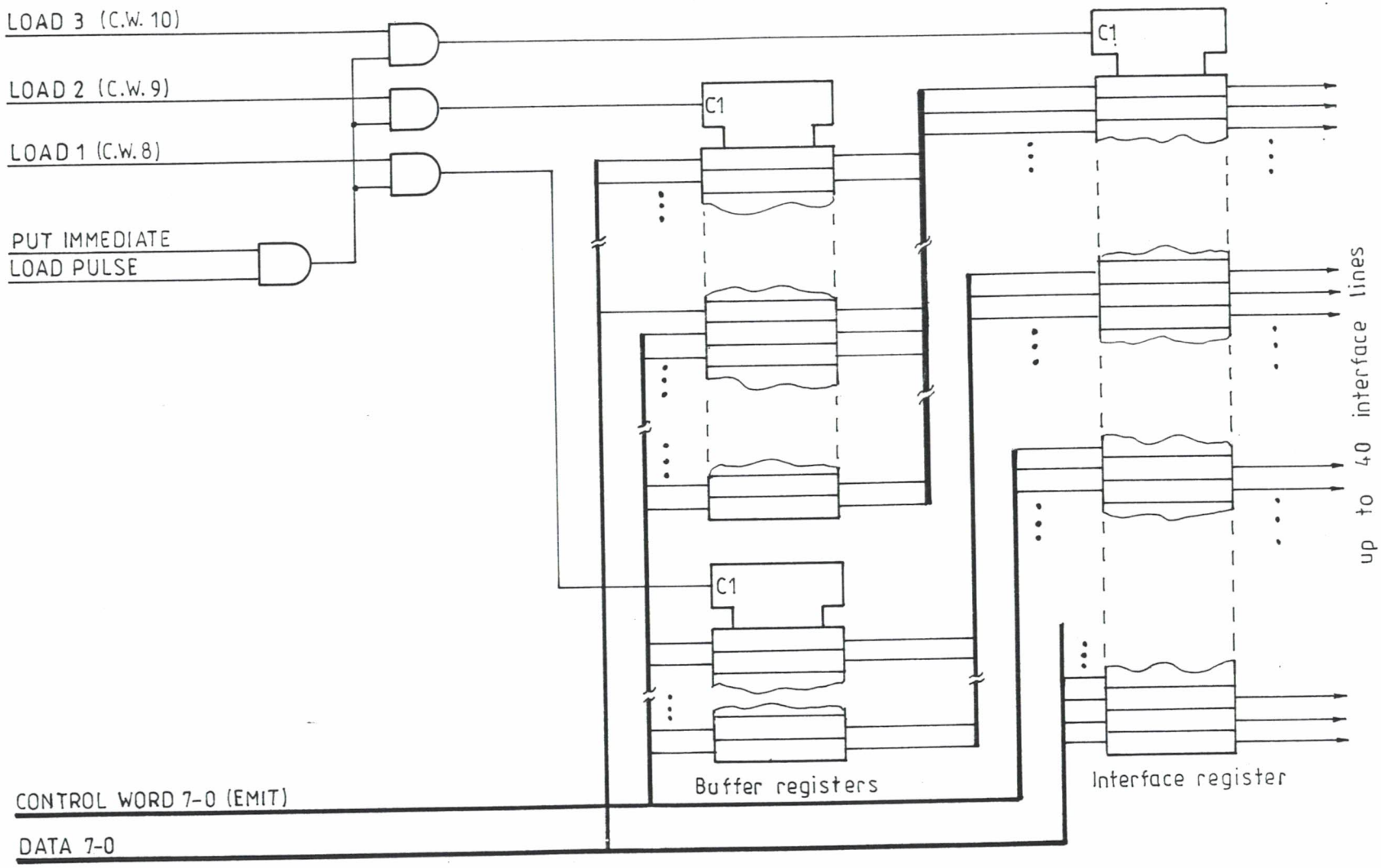

Figure 4: PUT IMMEDIATE hardware structure.

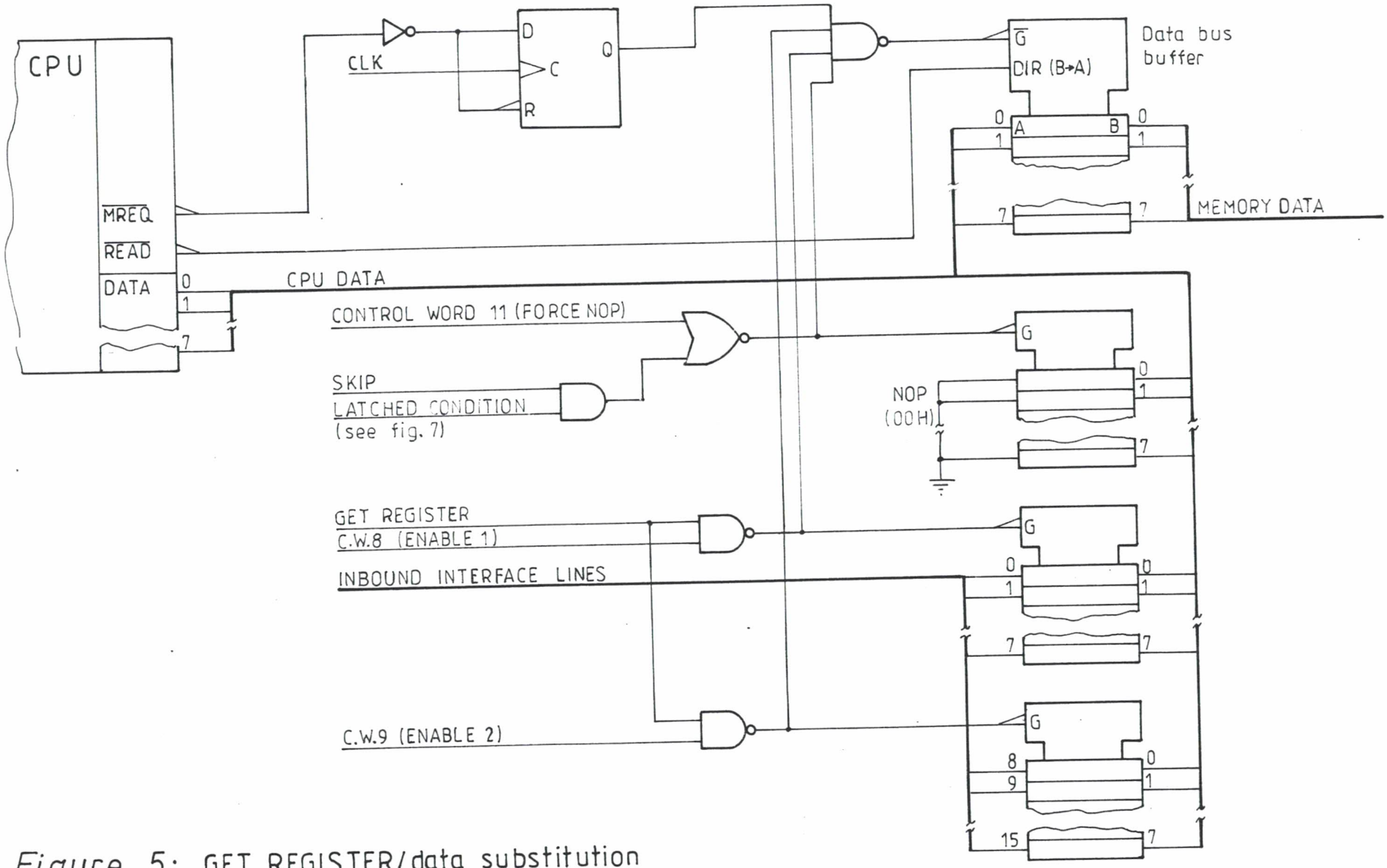

Figure 5: GET REGISTER/data substitution hardware structures:

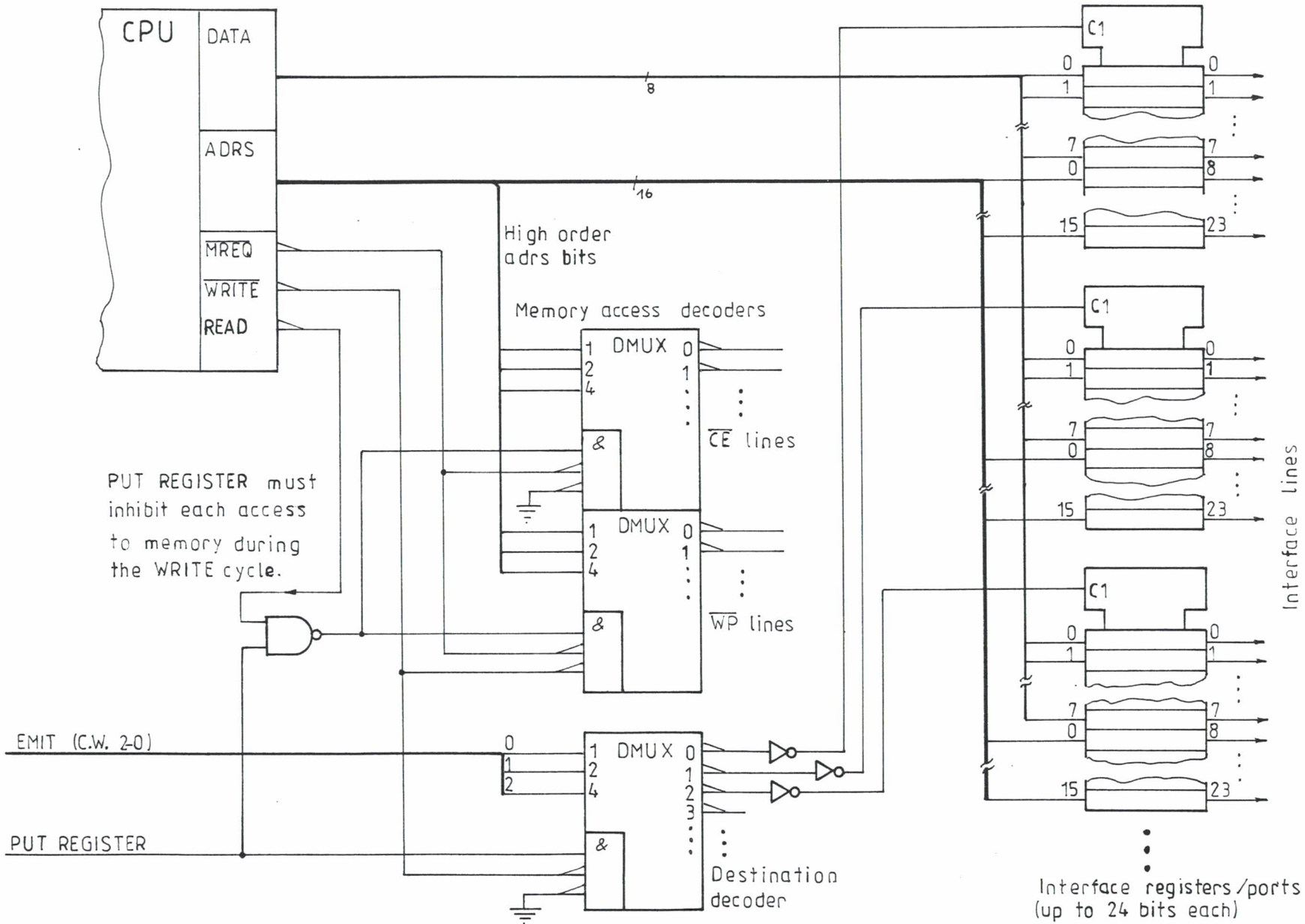

Figure 6: PUT REGISTER hardware structure.

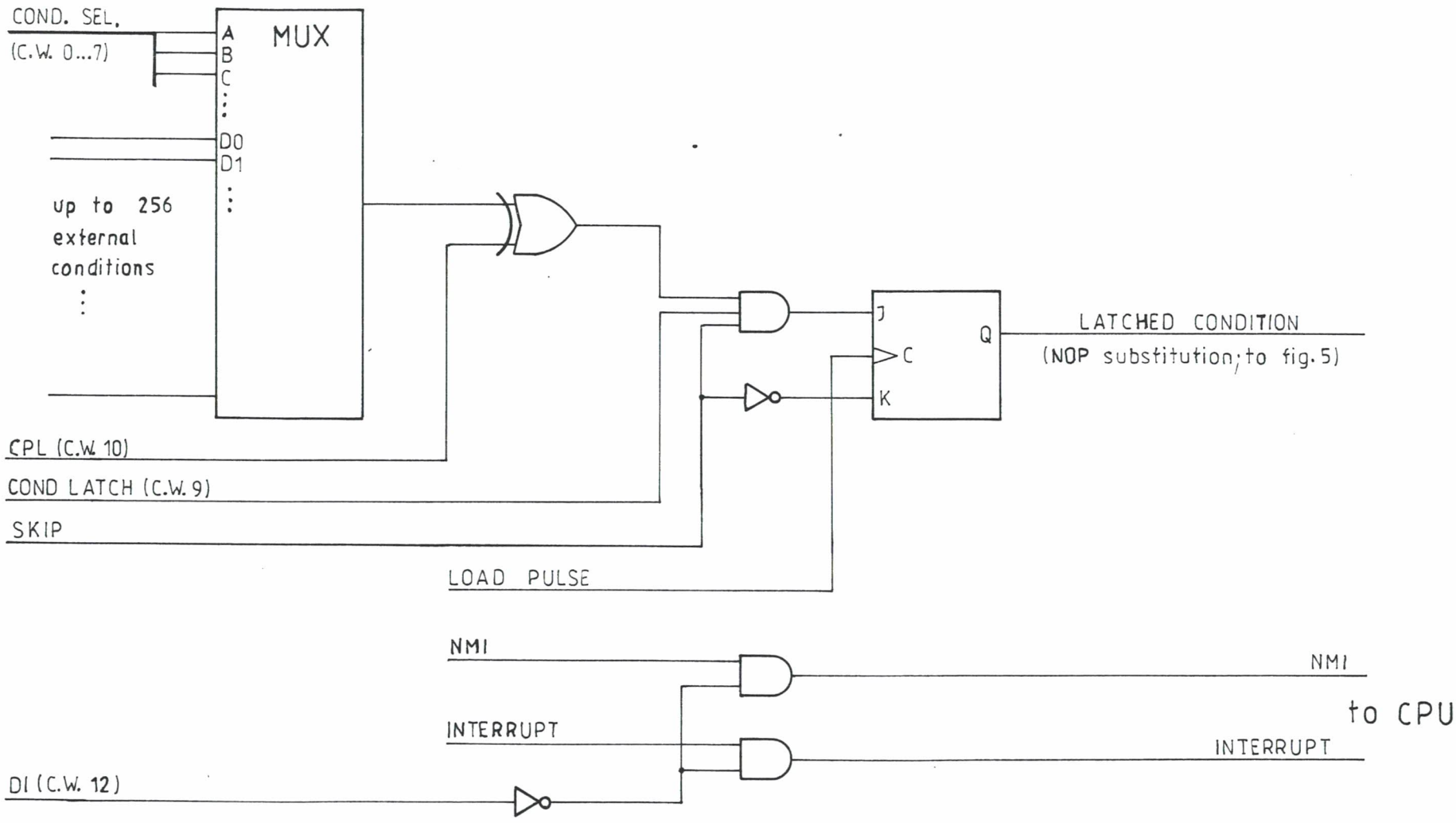

Figure 7: SKIP control circuitry.