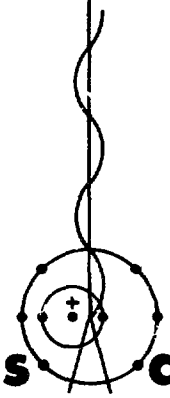
LA-5028-MS

INFORMAL REPORT

A Programmer's Guide to the Microprogrammed Branch Driver for the PDP-11



los alamos
scientific laboratory
of the University of California
LOS ALAMOS, NEW MEXICO 87544

UNITED STATES
ATOMIC ENERGY COMMISSION
CONTRACT W-7405-ENG. 36

LA-5028-MS
Informal Report
SPECIAL DISTRIBUTION
ISSUED: September 1972

los alamos
scientific laboratory
of the University of California
LOS ALAMOS, NEW MEXICO 87544

A Programmer's Guide to the Microprogrammed Branch Driver for the PDP-11

by

Philip R. Bevington*

*LASL Guest Scientist, MP-7
Present Address: Physics Department
Case Western Reserve University
Cleveland, Ohio

A PROGRAMMER'S GUIDE TO THE MICROPROGRAMMED BRANCH DRIVER FOR THE PDP-11

by

Philip R. Bevington

ABSTRACT

The Microprogrammed Branch Driver (MBD) for the PDP-11 computer originally designed by Buchanan and Jones[1] and since developed further by Biswell and Rajala[2] for the requirements of LAMPF has been described in some detail from the hardware point of view. This outline categorizes and describes the various components and instructions from the programmer's point of view to facilitate an understanding of the coding necessary for both the MBD processor and its associated PDP-11 computer. Stand-alone operation of the MBD is not covered. The version of MBD described is the LAMPF Mark II model; some of the mnemonics have been changed from their original definitions for consistency and compatibility with PDP-11 software.

1. DESCRIPTION OF REGISTERS

A. CPU Registers

Four registers are addressable by the computer Central Processor Unit (CPU) and control communication between the computer CPU and the processor of the Microprogrammed Branch Driver (MBD). All four are 16-bit registers with Device Addresses (DA) from 764000 to 764006.

1. CSR: Control and Status Register; DA 764000, write only (bits 0-6 and 8-10) or read only (bits 7 and 13-14).

Used to issue control functions such as "Reset MBD" or "Initialize Channel" and to indicate the status of various conditions. When the CSR is loaded with single-cycle mode specified (or bit 1 = Reset MBD), the MBD ceases all previous operations and executes the CSR function in one cycle and halts. When the CSR is loaded with bit 2 and run mode specified, the MBD continues with its current program (if any) to the next EXIT command, the Program Counter (PCR) of the MBD is then set to location 1, and the program stored in the Control Memory (CM) of the MBD is executed starting from location 1, with the active channel selected by bits 3-10 of the CSR. Note that channel interrupts from CAMAC will bypass this set of instructions which start at location 1 because such interrupts branch to location 0, which should contain a Jump Via CTR (JVC) instruction to branch to the beginning of the appropriate service routine.

Bit 0: Mode of operation - normal operation with a computer is in run mode; single-cycle mode is reserved for initialization and manual operation (0 = run mode, 1 = single-cycle mode).

Bit 1: Reset MBD - clears all channel request latches (CIL, CEL , PRL and CCL), sets the Program Counter (PCR) to 0, and sets the mode to single-cycle regardless of bit 0 of the CSR. The Ready bit (bit 7) of the CSR is set at the end of the MBD cycle. A Reset MBD instruction should precede the first Initialize Channel instruction to reset the PCR to 0.

Bit 2: Initialize Channel - bits 8-10 of the CSR set the Channel Initialize Latch (CIL) to the desired channel. The Ready bit (bit 7) of the CSR is set and the CIL is reset when the channel is accepted by the MBD, i.e., when the initialization routine for that channel begins at location 1. This

instruction increments the PCR to point to location 1, and it therefore assumes the PCR has been previously reset to Ø by a Reset MBD instruction from the computer CPU or by an EXIT instruction in the MBD program.

Bits 4-5: Extended Memory - not yet documented.

Bit 6: Interrupt Enable - enables the program interrupt (at priority BR5 with vectors starting at location 4ØØ in computer memory) to the computer CPU for the 25 interrupts of the MBD (Ø = disable, 1 = enable). The highest priority (at the highest location in computer memory 54Ø-543) is the branch or bus error, which is identifiable with bits 13 and 14 of the CSR; the next 8 interrupts are from XEQ INT commands from the 8 MBD channels; and the lowest priority interrupts are the low-order 16 bits of the Graded-Lam Requests (GLR) masked by the MASK (MSK) register.

Bit 7: Ready - status of MBD (bit 1) or initialized channel (bit 2); note that bit 7 is easily testable by byte tests for sign.

Bits 8-1Ø: Channel Select - selects which of 8 channels to initialize by bit 2.

Bit 13: Branch Error - a branch time-out error will generate an interrupt identifiable by this bit.

Bit 14: Bus Error - a UNIBUS transfer time-out error will generate an interrupt identifiable by this bit.

2. PDR: Program Data Register; DA 764ØØ2, read-write.

Used to transfer data or instructions between the computer CPU and the MBD registers or Control Memory (CM).

3. MSK: MaSK for GLR interrupts; DA 764ØØ4, read-write.

Used to mask (AND) the low-order bits (1-16) of the Graded Lam (GL) interrupt requests (GLR). A bit = 1 will enable an interrupt at the corresponding priority.

4. LIR: Load Instruction Register; DA 764ØØ6, write only.

Used to load the Instruction Register (IR) of the MBD from the computer CPU and execute the instruction. When the LIR is loaded, the contents are transferred to the IR and the instruction is executed; the mode should be single-cycle for proper operation.

B. DMA Registers

Two 16-bit registers are addressable only by the MBD but control the flow of information on the UNIBUS.

1. MAR: Memory Address Register.

Contains the address in computer memory for Direct Memory Access (DMA) transfers. This address is a word address, whereas addresses transferred from PDP-11 programs are generally byte addresses and must be divided by two (or shifted right one bit).

2. MDR: Memory Data Register.

Contains the data of the last DMA transfer from memory or the next transfer to computer memory.

C. CAMAC Registers

Three registers are addressable by the MBD and control communication between the MBD and the CAMAC crates and modules.

1. BAR: Branch Address Register; 16 bits decoded into 21 bits.

Used to specify the Crate, Number, Address, and Function (CNAF) command for CAMAC operation.

Bits Ø-3: Address (A) or register within module; transferred as 4 bits BA1, 2, 4, 8 to CAMAC.

Bits 4-8: Number (N) of module within crate; transferred as 5 bits BN1, 2, 4, 8, 16 to CAMAC.

Bits 9-11: Crate number (C) between 1 and 7; decoded to one bit of 7 bits BCR1-BCR7 for CAMAC with one bit per crate.

Bits 12-15: Function (F) to be performed, not including F8, which is specified by the control (CTRL) portion of the Instruction Register (IR) by BRN (F8 = Ø) or BRC (F8 = 1); decoded into 5 bits BF1, 2, 4, 8, 16, including F8, to CAMAC.

2. BDR and BDH: Branch Data Register; 24-bit register divided into two parts for manipulation by 16-bit processor.

BDR: Low-order bits (1-16) of CAMAC data.

BDH: High-order bits (17-24) of CAMAC data.

3. GLR: Graded L's Register: 24-bit register of which only the low-order portion is addressable by the MBD.

GLR: Low-order bits (1-16) of the Graded Lam (GL) requests; computer CPU program interrupts from these bits are masked with MSK.

GL Channel Requests: High-order bits (17-24) of the GL requests are interpreted directly as requests to the corresponding 8 channels (Ø-7) of the MBD, with channel 7 highest priority and channel Ø lowest priority.

D. File Registers

Each of the 8 channels of the MBD has 2 banks of 7 registers (all 16-bit registers) to control the operation of each channel (14 registers per channel). The Channel Control Latch (CCL) determines which set of file registers (which channel) is active. Selection of banks is made by the appropriate ConTRoL (CTRL) function in a normal instruction. Normal operation is through the registers of bank Ø, which is selected by any EXIT operation (so that each interrupt begins in bank Ø); the registers of bank 1 are available for auxiliary operation. Five of the registers are nominally assigned specific purposes, but only the CTR is restricted to specific functions; all may be specified as the SouRCe (SRC) or DeSTination (DST) of a normal instruction.

1. CTR: ConTrol Register.

Contains status bits and the program address in Control Memory (CM) for interrupts and addressing the CM by normal instructions. Bits Ø-11 of the CTR may be loaded directly from the Instruction Register (IR) by an LCI instruction as well as being specified as the SRC or DST of a normal instruction.

Bits Ø-11: Program address in CM used by normal instructions with CM as SRC or DST or by Jump Via CTR (JVC) instruction to begin processing an interrupt at the specified address; high-order bits are ignored for CM less than 4Ø96 words.

Bits 12-15: Status bits which may be set and tested by MBD instructions.

2. ILR: Instruction List Register.

List pointer which contains the address of the next word in the list of instructions stored in the computer memory; used to transfer instructions from the computer memory to the MBD processor.

3. DAR: Data Address Register.

List pointer which contains the address of the list of data stored in computer memory, used to transfer data between the MBD and computer memory.

4. WCR: Word Count Register.

Contains the running word count of the number of words to be transferred in a block transfer of data.

5. CCR: CAMAC Command Register.

CAMAC CNAF command as for the BAR with 16 bits, not including F8.

6. GP1: General Purpose Register Number 1.

For general use.

7. GP2: General Purpose Register Number 2.

For general use. Note that the 7 auxiliary registers in bank 1 can be used as additional general-purpose registers (though addressed as specific file registers). The CTR in bank 1 functions in the same way as the CTR in bank Ø.

E. MBD Registers

A number of special-purpose registers are addressable in whole or in part by the MBD processor.

1. CM: Control Memory.

Nominally 1Ø24 words (minimum 256, maximum 4Ø96) of 16-bit memory containing program instructions and data for MBD operations. Addressable either by the STOre (STO) and LOaD (LOD) instructions or as the SouRCe (SRC) or DeSTination (DST) of a normal instruction.

2. IR: Instruction Register.

16-bit register which contains the instruction to be executed by the MBD. The IR may be loaded from the computer CPU by loading the LIR, after which the instruction is executed in single-cycle mode. In normal run-mode operation, the IR is loaded automatically in sequence from the address in the CM specified by the Program Counter (PCR).

3. ATR: ArIThmetic Register.

16-bit register which contains the result of the last normal operation; its value is included as a source in many of the normal instruction operations. The SHiFt (SHF) instruction operates directly on the ATR without using the arithmetic logic unit of the MBD processor, so that the tests for zero and carry are not valid, though the test for negative is. For all other instructions the

ATR and the arithmetic logic unit can be considered synonomous.

4. PCR: Program Counter Register.

12-bit register pointing to the address in the CM of the next instruction to be executed. The PCR is set to Ø by the Reset MBD function (bit 1) of the CSR or by any EXIT command, loaded from the channel CTR by a JVC instruction, incremented by the Initialize Channel function (bit 2) of the CSR or by execution of any IR instruction in single-cycle mode, incremented automatically during normal instruction sequencing, set by the JVA instruction, or set or incremented by branch instructions BCT or BCF. It may also be read as a SouRCe (SRC) of a normal instruction, but not altered by writing as a DeSTination (DST). Note that the PCR of the MBD points to word addresses whereas the PC of the PDP-11 computer CPU points to byte addresses.

5. CIL: Channel Initialize Latch.

8-bit register (of which only one bit may be set at a time) used to initialize any one of the 8 channels. It is loaded from bits 8-1Ø of the CSR (with bit 2 set) or with the CIL SET Command and is reset by any EXIT command in that channel, or by a Reset MBD function (bit 1) of the CSR.

6. CEL: Channel Enable Latch.

8-bit register (one bit per channel) specifying which channels have hardware interrupts enabled. An EXIT 2 command will selectively set (enable) the appropriate bit of the CEL corresponding to the channel of the top 8 bits of the GL (if CAMAC) or the channel being initialized (if computer CPU); an EXIT 3 or EXIT 4 command selectively resets (disables) the corresponding bit.

7. PRL: Program Request Latch.

8-bit register (one bit per channel) specifying which channels have programs waiting to be executed. An EXIT 1 command will set the appropriate bit in the PRL corresponding to the channel initiating the interrupt being serviced (if CAMAC) or the channel being initialized (if computer CPU); an EXIT 2 or EXIT 4 command selectively resets the corresponding bit. Setting a bit in the PRL is equivalent to issuing a hardware interrupt request to the corresponding channel with the channel enabled; when no higher priority channels are active or requesting service, that channel is selected and the instruction (JVC) in location Ø of the CM is executed to branch to the service routine for that channel.

8. CCL: Channel Control Latch.

8-bit register (of which only one bit may be set at a time) specifying which channel is currently in control and active. The CCL may be read as a SRC, but not altered by software as a DST. The CCL is set to a specific channel whenever that channel is being initialized by the computer CPU (by loading the CSR with bit 2 and the channel number in bits 8-1Ø) or a hardware interrupt to that channel (from CAMAC GL bits 17-24) is accepted and service begun. The CIL can be loaded with a CIL SET command, and the channel number loaded into the CIL will be in the priority request at the next EXIT command; this is an indirect load of the CCL, which is required for stand-alone systems in order to change channels.

II. DESCRIPTION OF INSTRUCTIONS

A. Normal Instructions

Normal instructions are comprised of four parts: the OPeration (OP) is specified by bits 12-15, the SouRCe (SRC) and DeSTination (DST) registers are specified by bits 4-7 and Ø-3, respectively, and an optional ConTRoL (CTRL) command is specified by bits 8-11. The SRC and/or DST can be omitted (NON = ØØ) for operation with the ATR; the result of any operation (by a normal instruction) is always left in the ATR as well as the DST. The octal representations of all instructions and elements are given in Table I; a complete instruction is the sum of the appropriate codes, one from each column. Some mnemonics have been changed from their original definitions for consistency and compatibility with PDP-11 software; SRC and DST mnemonics may be distinguished by adding prefexes of S and D, respectively. If the SRC or DST is Control Memory (CM), then the address is in bits Ø-11 of the CTR.

1. MVE: MoVE SRC to DST and execute CTRL.

2. INM: INcreMent SRC, store the result in both SRC (only if it is a file register and not the CTR) and DST, and execute CTRL. The CARRY flag is set on overflow from 177777 to Ø.

3. DEM: DEcreMent SRC, store the result in both SRC (only if it is a file register and not the

TABLE I

OCTAL REPRESENTATION OF INSTRUCTIONS

Operation		Control		Condition		Source		Destination	
000000	MVE	NONE	0000	BRD	000	NON	00	ATR
010000	INM	MRDR	0400	DCH BSY	020	ILR	01	ILR
020000	DEM	MWTR	1000	BR BSY	040	DAR	02	DAR
030000	ADT	MRDH	1400	INT BSY	060	WCR	03	WCR
040000	SBT	MWTH	2000	ATR 15	100	CCR	04	CCR
050000	IOR	BRN	2400	ZERO	120	CTR	05	CTR
060000	XOR	BRC	3000	ZERO LO	140	GP1	06	GP1
070000	AND	EXIT 4	3400	CARRY	160	GP2	07	GP2
100000	BCT	EXIT 1	4000	Q FLAG	200	PCR	10	CLR
110000	BCF	EXIT 2	4400	X FLAG	220	MDR	11	MDR
120000	JVC	EXIT 3	5000	ATR 00	240	MAR	12	MAR
130000	STO	XEQ INT	5400		250	BDR	13	BDR
140000	LOD	CIL SET	6000	CTR 12	300	BDH	14	BDH
150000	SHF	BANK 0	6400	CTR 13	320	SCL	15	BAR
160000	LCI	BANK 1	7000	CTR 14	340	PDR	16	PDR
170000	JVA	EZ COM	7400	CTR 15	360	CM	17	CM

030000 DEP Deposit ATR into DST (ADT with no SRC)

0000 LEFT Direction for SHF (bit 10) Shift left

2000 RIGHT Direction for SHF (bit 10) Shift right

0000 CIR Type for SHF (bit 11) Circular shift

4000 NRM Type for SHF (bit 11) Normal shift

150000 SCL Shift Circular Left

152000 SCR Shift Circular Right

154000 SNL Shift Normal Left

156000 SNR Shift Normal Right

3 MBDRST Reset MBD (CSR function) Bit 1 (and bit 0)

4 CHINIT Channel Initialize (CSR function) Bit 2

CTR) and DST, and execute CTRL. The CARRY flag is set on ADT underflow from 0 to 177777.

4. ADT: ADd 2's complement ATR to SRC, store the result in DST, and execute CTRL. The CARRY flag is set on overflow if SRC + ATR > 177777.

5. SBT: SuBTract ATR from SRC, store the result in DST, and execute CTRL. The CARRY flag is set if SRC ≥ ATR.

6. IOR: Inclusive OR ATR and SRC, store the result in DST, and execute CTRL.

7. EOR: EXclusive OR ATR and SRC, store the result in DST, and execute CTRL.

8. AND: AND ATR and SRC, store the result in DST, and execute CTRL.

9. DEP: DEPosit ATR into DST and execute CTRL. Any one of three instructions (ADT, IOR, or EOR) when used without a SRC.

B. Branch-Conditional Instructions

Branch-on-condition instructions are comprised of three parts: the OP is specified by bits 12-15, the CONDition (COND) to be tested is specified by bits 8-11, and the ADDRess (ADDR) in the CM for the branch is specified by bits 0-7. Since the maximum address specified is 8 bits, branching may only be done within the current page of 256 words. The ATR and conditional flags are not effected by branch instructions.

1. BCT: Branch to ADDR if COND is True - Continue with PCR = PCR + 1 if COND false.

2. BCF: Branch to ADDR if COND is False - Continue with PCR = PCR + 1 in COND true.

C. CM Addressing Instructions

The Control Memory (CM) addressing instructions are comprised of two parts: the OP is specified by bits 12-15, and the ADDRess (ADDR) in the CM is

specified by bits Ø-11 (except for the JVC and LCI instructions).

1. STO: STOre ATR into CM at ADDR; if mode is single-cycle, the source is the PDR rather than the ATR.

2. LOD: LOaD the contents of ADDR in CM into ATR; if mode is single-cycle, the destination is the PDR rather than the ATR, and the ATR is not affected.

3. JVA: Jump Via ADDR - deposit ADDR into the PCR, effecting a jump to the address specified by the IR. The ATR is not affected.

4. JVC: Jump Via CTR - transfer bits Ø-11 of the appropriate CTR (specified by the CCL either by the interrupting channel or the channel being initialized) into the PCR, effecting a jump to the address specified in the CTR of the appropriate channel. Note that JVC should be the first instruction of the MBD program (in location Ø) so that hardware interrupts, which branch to location Ø, can jump to the service routine appropriate for the interrupting channel.

5. LCI: Load CTR from IR - transfer bits Ø-11 of the IR to the CTR, providing a pointer for interrupts to the address specified by the IR. Note that the LCI instruction does not directly address the CM or affect the PCR, but prepares the CTR for eventual use by the PCR, or by a normal instruction with the CM as a SRC or DST. It does affect the ATR, loading it with bits Ø-11 of the IR.

D. Shift Instruction

The shift instruction is comprised of three parts: the OP is specified by bits 12-15, the number of bits to be shifted is specified by bits Ø-3, and the direction and type of shift are specified by bits 1Ø and 11.

1. SHF: SHiFt ATL N bits right/left and normal/circular.

Bits Ø-3: N (Ø-15) - number of bits ATL is to be shifted.

Bit 1Ø: Direction - Ø = left, 1 = right.

Bit 11: Type - Ø = circular (rotate), 1 = normal (logical).

E. Control Commands

The CouTRoL command (CTRL) is specified by bits 8-11 of a normal instruction. It is normally executed as part of (and after in timing) a normal instruction. Note that an instruction consisting of only a CTRL command is essentially a MVE NON, NON, CTRL instruction and its only effect besides the CTRL execution is to clear the ATL; the ATL is unaffected if the SRC and DST are left blank for ADT, IOR, EOR, or AND OP codes.

1. MRDR: Computer Memory Read and Release.

Read one word from computer memory at MAR into MDR and release the UNIBUS after transfer is complete.

2. MWTR: Computer Memory WriTe and Release.

Write one word from MDR into computer memory at MAR and release the UNIBUS after transfer is complete.

3. MRDH: Computer Memory ReaD and Hold.

Read one word from computer memory at MAR into MDR and hold the UNIBUS (retain control as master) after transfer is complete. The UNIBUS can be released by a MRDR or MWTR instruction or by any EXIT command.

4. MWTH: Computer Memory WriTe and Hold.

Write one word from MDR into computer memory at MAR and hold the UNIBUS (retain control as master) after transfer is complete.

5. BRN: BRanch Normal Command.

Issue to CAMAC the CNAF command stored in the BAR with F8 = Ø (normal read/write commands).

6. BRC: BRanch Control Command.

Issue to CAMAC the CNAF command stored in the BAR with F8 = 1 (control commands).

7. BZ COM: Branch Zero COMmand.

Issue to CAMAC the BZ command to clear and initialize. The hardware reset of the PDP-11 clears the CAMAC branch with a BZ command, but does not reset the MBD.

8. EXIT 1: EXIT at priority 1.

Suspend program execution for the channel in progress and set the corresponding bit of the PRL. When no higher priority channels are active or requesting service, that channel is selected again and the PCR is set (by a JVC instruction at location Ø) to the location in CM specified by bits

Ø-11 of the CTR of that channel. Used to relinquish control temporarily to higher priority channels if they are requesting service.

The EXIT commands provide an opportunity for interrupt requests to be serviced. While a program is running in the MBD it cannot be interrupted, except by a Reset MBD command from the computer CPU. Any EXIT command terminates program execution, resets the PCR to Ø, selects bank Ø, examines the priority of all interrupt requests, and begins to service the interrupt request from the channel with the highest priority.

If a program running at low priority wants to allow higher-priority requests to interrupt, it may issue an EXIT 1 command (after setting the CTR to point to the next instruction), which sets the PRL bit representing a software request to interrupt at that channel. Execution of the program will be suspended temporarily; as soon as all (if any) higher-priority channel requests (hardware or software) have been satisfied, execution of the program will continue (at the location specified by the CTR).

When a program is finished, but more hardware requests on that channel are expected, an EXIT 2 command will set the CEL bit to enable CAMAC Graded-Lam requests (GL) on that channel to request an interrupt; the PRL bit is reset so that no request appears on that channel until the hardware request from the GL. An EXIT 4 command prevents any request on that channel from appearing until the channel is re-initialized. An EXIT 3 command disables hardware (CEL) requests, but leaves the software (PRL) requests unchanged from their previous settings by EXIT 1 or EXIT 2 commands.

9. EXIT 2: EXIT at priority 2.

Suspend program execution for the channel in progress, reset the corresponding bit of the PRL, and set the corresponding bit of the CEL. Program execution for that channel will commence at the address specified in bits Ø-11 of the CTR for that channel when a hardware interrupt on the corresponding bit from the CAMAC GL (bits 17-24) is requested and no higher priority channels are active or requesting service. Used at the end of an interrupt servicing routine.

10. EXIT 3: EXIT at priority 3.

Suspend program execution for the channel in progress and reset the corresponding bit of the CEL. Interrupt requests from the CAMAC GL (bits 17-24) will be ignored. If the corresponding bit of the PRL is already set (by a previous EXIT 1), EXIT 3 will act like EXIT 1; otherwise it will act like EXIT 4. Used to disable interrupts to a channel.

11. EXIT 4: EXIT at priority 4.

Suspend program execution for the channel in progress and reset the corresponding bit of both the PRL and the CEL. Program execution for that channel is terminated until that channel is re-initialized by the computer CPU.

12. XEQ INT: eXEQute interrupt.

Issue a program interrupt to the computer CPU at the interrupt vector associated with the channel in progress (as determined by the CCL). The 8 channels (Ø-7) correspond to high priority interrupts (17-24); only one channel can have an interrupt executed at one time.

13. CIL SET: Channel Initialize Latch SET.

Set the appropriate bit of the CIL from the ATR; pertains to stand-alone operation only. The instruction MVE SRC, CILSET loads the CIL with the number in bits Ø-2 of the SRC.

14. BANK Ø: specify BANK Ø file registers.

Normal operation of the special-purpose file registers (CTR, ILR, DAR, WCR, and CCR) is with the registers of bank Ø; this set may be specified by either the BANK Ø or any EXIT control command. Hardware interrupts may assume bank Ø because the last command before an interrupt is accepted for service can only be an EXIT command.

15. BANK 1: specify BANK 1 file registers.

Each channel has an auxiliary set of 7 file registers with the same SRC and DST names as those of bank Ø, but which are available for any purpose. Note that instructions which operate on the CTR (JVC and LCI) will operate equally well on those of bank 1, as well as those of bank Ø, depending on which bank is operative.

F. Condition Tests

The CONDition test (COND) is specified by bits 8-11 of a branch-conditional instruction (BCT or BCF). Note that the tests for zero and carry

(ZERO, ZERO LO, and CARRY) are not operative (always reset) after a SHF instruction, which operates directly on the ATR without using the arithmetic logic unit. A null instruction after the shift (e.g., ADT, IOR, or EOR with no SRC or DST) will set the zero flags properly for testing, but a test for carry after a SHF instruction is not possible.

1. DCH BSY: Data CHannel BuSY.

True if the UNIBUS data channel is busy.

2. BR BUSY: BRanch BUSY.

True if the CAMAC branch line is busy.

3. INT BSY: INTerrupt BuSY.

True if a computer CPU interrupt is already present and not yet satisfied.

4. BRD: CAMAC BRanch Demand.

True if a CAMAC interrupt request is present.

5. ZERO: ATR ZERO word.

True if the entire word of the ATR is zero.

6. ZERO LO: ATR ZERO LOw byte.

True if the low byte of the ATR is zero.

7. CARRY: arithmetic CARRY.

True if the last arithmetic operation resulted in a carry (overflow). See the description of normal instructions for details.

8. Q FLAG: Q FLAG of CAMAC.

True if a Q response was generated by the last CAMAC command issued.

9. X FLAG: X FLAG of CAMAC.

True if an X response was generated by the last CAMAC command issued.

10. ATR ØØ: ATR bit ØØ.

True if bit Ø of the ATR is true.

11. ATR 15: ATR bit 15.

True if bit 15 of the ATR is true.

12. CTR 12 - CTR 15: CTR bits 12-15.

True if the corresponding bit of the channel CTR is true. Bits 12-15 of the CTR may be set under program control (e.g., with an instruction like MVE,SRC, CTR with the appropriate bits set in the SRC register).

III. ILLUSTRATIVE EXAMPLE PROGRAMS

A. Initialize MBD and CAMAC

Initialization of the MBD processor and the CAMAC branch is performed in single-cycle mode. The Reset MBD function of the CSR takes precedence over all MBD functions and causes it to cease operation at the end of the current MBD cycle. Branch and CNAF commands can be used to initialize the CAMAC controllers and modules. The process is illustrated in Program 1. Note that MBD instructions (e.g., MVE+SPDR+DBAR+BRC = MVE PDR,BAR,BRC) are constructed by adding the appropriate mnemonics, which must be defined for the assembler as in Program I, or by defining all the mnemonics of Table I permanently. (The SRC and DST values are differentiated by S and D prefixes in this illustration.) Note also the use of @# to generate absolute addresses.

1. Reset MBD.

Loading the CSR with the Reset MBD command (bit 1) halts and resets the MBD processor.

2. Clear Branch.

The BZ (clear and initialize) CAMAC command can be issued to the CAMAC branch by loading and executing a BZ COM instruction through the LIR (in single-cycle mode). CAMAC requires a delay of 15 microseconds before the next branch command.

3. Enable Interrupts.

The mask (MSK) for the low-order GL interrupt bits (GLR) may be cleared or selectively set directly by the CPU. A list of CNAF commands to enable or disable interrupts can be issued by loading them through the PDR to the BAR (by loading and executing the LIR). Inclusion of a BRC (or BRN) CTRL command issues the CNAF command in the BAR. The following CNAF commands are generally useful:

1Ø1751	C1, N3Ø,	A9, F24	= turn inhibit (I) off.
121751	C1, N3Ø,	A9, F26	= turn inhibit (I) on.
1Ø1752	C1, N3Ø,	A1Ø, F24	= disable branch demand (BD) for GL.
121752	C1, N3Ø,	A1Ø, F26	= enable branch demand (BD) for GL.

B. Load Programs into CM

The programs for the MBD processor must be written in MBD language, but included in the loading PDP-11 computer program. A short program to transfer the proper code from the computer to the MBD is

Program I

Code to Initialize CAMAC

```
CSR=764ØØØ                              ;CSR (CPU REGISTER)
BZCOM=74ØØ                              ;BZ COM (CTRL)
LIR=764ØØ6                              ;LIR (CPU REGISTER)
MSK=764ØØ4                              ;MSK (CPU REGISTER)
PDR=764ØØ2                              ;PDR (CPU REGISTER)
MVE=Ø                                   ;MVE (OP)
SPDR=34Ø                                ;SRC PDR (FILE REGISTER)
DBAR=15                                 ;DST BAR (CAMAC REGISTER)
BRC=3ØØØ                                ;BRC (CTRL)
MBDINZ: MOV   #3,@#CSR                  ;RESET MBD (CSR FUNCTION BIT 1)
        MOV   #BZCOM,@#LIR              ;ISSUE BZ COMMAND (CLEAR CAMAC)
        MOV   #1Ø,RØ                    ;WAIT 15 MICROSECONDS AFTER BZ
MBD11:  DEC   RØ
        BGT   MBD11
        MOV   @#MASK,@#MSK              ;SET SELECTED BITS OF GLR MASK
        MOV   #FNABGN,RØ                ;FNABGN = BEGINNING OF CNAF COMMAND LIST
MBD12:  MOV   (RØ)+,@#PDR               ;TRANSFER CAMAC SETUP COMMANDS
        MOVE  #MVE+SPDR+DBAR+BRC,@#LIR  ;ISSUECNAF COMMAND (F8=1)
        CMP   RØ,#FNAEND                ;CHECK FOR END OF CNAF COMMAND LIST
        BLO   MBD12                     ;CONTINUE CAMAC SETUP
        BR    MBDLOD                    ;GO TO NEXT PROGRAM
MASK:   177ØØØ                          ;MASK BITS 9-16 OF GLR INTERRUPTS ON
FNABGN: 1Ø1751                          ;F24 C1 N3Ø A9 = TURN INHIBIT OFF
        121752                          ;F26 C1 N3Ø A1Ø = ENABLE BD (GL)
FNAEND:
```

Program II

Code to Load Instructions from Compuber Memory to CM

```
MDBLOD: MOV   #MBDBGN,R1                ;MBDBGN = BEGINNING OF MBD PROGRAM LIST
        MOV   #STO,RØ                   ;STO = STORE PDR INTO CM AT ADDR=Ø
MBD21:  MOV   (R1)+,@#PDR               ;STORE PROGRAM STARTING FROM LOCATION Ø
        MOV   RØ,@#LIR                  ;LOAD AND EXECUTE STO INSTRUCTION
        INC   RØ                        ;POINT ADDR TO NEXT LOCATION IN CM
        CMP   R1,#MBDEND                ;CHECK FOR END OF PROGRAM LIST
        BLO   MBD21                     ;CONTINUE STORING
        MOV   #MBDBGN,R1                ;CHECK STORED PROGRAM FOR ERRORS
        MOV   #LOD,RØ                   ;LOD = LOAD CM AT ADDR=Ø INTO PDR
MBD22:  MOV   (RØ),@#LIR                ;READ PROGRAM IN CM INTO PDR
        CMP   (R1)+,@#PDR               ;COMPARE WITH ORIGINAL IN COMPUTER MEMORY
        BNE   MBDLOD                    ;LOAD AGAIN IF ERROR FOUND
        INC   RØ                        ;POINT ADDR TO NEXT LOCATION IN CM
        CMP   R1,#MBDEND                ;CHECK FOR END OF PROGRAM
        BLO   MBD22                     ;CONTINUE CHECKING
```

illustrated in Program II. The entire programming for the MBD is assumed to be stored in the computer program between MBDBGN and MBDEND.

1. Load Program.

The program is loaded word by word through the PDR. The STO instruction with an initial address of ADDR=Ø is loaded into the LIR to effect the transfer to the location at ADDR in the CM. The pointer (R1) to code in the computer is incremented (1 location = 2 bytes) in synchronization with the address (1 location = 1 word) of the STO instruction (RØ).

2. Check Program.

The program can be read back through the PDR with a LOD instruction in the LIR to check for errors.

C. Initialize Selected Channels

Initialization of channels requires coordination of programs in both the computer and the MBD (e.g., Programs III and IV). The computer should supply (for each channel) pointers to the service routine in the CM and to lists of constants. The MBD program should retrieve these pointers and store them appropriately.

1. CPU Program.

The CPU program is illustrated in Program III. A list of pointers begins at CHNBGN with two pointers per valid channel and one Ø for each channel which is not to be initialized. The first word points to the service routine in the CM and the second word points to a list of constants stored elsewhere in the computer program. A pointer to the appropriate pair of pointers is loaded into the PDR. Loading the CSR with bit 2 and the channel number in bits 8-1Ø begins initialization. Bit 7 of the CSR indicates when the channel is accepted and initialization of that channel has begun in Program IV in the MBD; a time-out loop checks for malfunction of a channel (lack of an EXIT from the previous channel program). While one channel is being initialized by Program IV, the next is being readied in Program III. After initialization, the CSR (bits 13-14) may be checked for errors.

2. MBD Program.

The MBD program is illustrated in Program IV. The code begins at location Ø with a JVC instruction to link interrupts to their own service routines. Initialization begins at location 1, with a (byte) pointer in the PDR to constants in the computer memory. Afterwards, the PDR should not be accessed by the MBD because the CPU may change it. Two words are read in through the MDR from the (word) addresses specified by the MAR, and deposited in the CTR and LIR for later use. Bit 15 of the CTR is checked to see whether to continue initialization and/or execution via an EXIT 1 command. Note that the MBD addresses begin at location Ø whereas the PDP-11 addresses are relocatable; the BCT instructions, for example, branch to their own addresses (location 4 or 7) to execute wait loops. Note also that MBD addresses, including the MAR, are word addresses, whereas PDP-11 code produces byte addresses.

D. Display Spectrum

A "typical" MBD program to display on a CAMAC-oriented scope a spectrum stored in computer memory is illustrated in Program V. The file registers are loaded from a list of constants in computer memory (with some manipulation for GP1 and GP2). The X position in the GP1 is incremented uniformly by GP2 and loaded directly to the display module. The Y position is scaled by a SHF instruction, which is constructed from initial constants, and loaded with the intensification command. An EXIT 1 command relinquishes control to higher-priority channels after each point is displayed. Note that the execute commands and wait loops for the data channel and CAMAC branch are separated as much as possible (within the display loop) to allow the transfer to overlap calculations.

E. Data Acquisition

A minimal program to read data from CAMAC registers and store them in computer memory is illustrated in Program VI. A block of data is read from the CAMAC registers by executing a stored list of CAMAC CNAF commands, and the data are transferred simultaneously to the computer memory. The registers may be read in any order, as specified by the stored list of CNAF commands. A pointer to the buffer in the computer memory is assumed to be stored in the ILR; the procedure of Programs III and IV may be used to load the beginning address into the ILR initially, and to reset the pointer to the beginning of the buffer periodically. The value of the pointer

Program III

CPU Code to Initialize Channels

```
MBDCHI: MOV     #3,@#CSR        ;RESET PC TO Ø
        MOV     #CHNBGN,R1      ;CHNBGN = BEGINNING OF LIST OF POINTERS
        MOV     #4,RØ           ;CHANNEL INITIALIZE (BIT 2 OF CSR)
MBD31:  MOV     R1,@#PDR        ;LOAD POINTER TO DATA LIST INTO PDR
        TST     (R1)+           ;EXAMINE FIRST WORD
        BEQ     MBD34           ;OMIT CHANNEL IF ADDRESS IS Ø
        MOV     RØ,@#CSR        ;INITIALIZE CHANNEL
        TST     (R1)+           ;INCREMENT LIST POINTER
        MOV     #1ØØØ.,R2       ;WAIT FOR MBD INITIALIZE PROGRAM
MBD32:  TSTB    @#CSR           ;TEST READY BIT 7 OF CSR
        BMI     MBD34           ;CONTINUE INITIALIZATION WHEN READY
        DEC     R2              ;WAIT FOR ABOUT 10 MSEC MAXIMUM
        BGT     MBD32           ;HALT AFTER 10 MSEC
MBD33:  HALT                    ;HALT WITH CHANNEL NUMBER IN RØ
        BR      MBDCHI+1        ;BEGIN INITIALIZATION AGAIN
MBD34:  ADD     #4ØØ,RØ         ;INCREMENT CHANNEL NUMBER
        CMP     R1,#CHNEND      ;CHECK FOR END OF POINTER LIST
        BLO     MBD31           ;CONTINUE WITH NEXT CHANNEL
        MOV     @#CSR,RØ        ;CHECK FOR BRANCH OR BUS ERROR
        BIT     #6ØØØØ,RØ       ;BITS 13 AND 14 of CSR
        BNE     MBD33           ;HALT IF ERROR FOUND
        JMP     MAIN            ;GO TO MAIN PROGRAM
CHNBGN: CHØBGN+1ØØØØØ           ;BEGINNING IN CM OF CODE FOR CHANNEL Ø
        CHØLST                  ;BEGINNING IN COMPUTER MEMORY OF LIST
        Ø,Ø,Ø,Ø,Ø,Ø             ;ONE Ø FOR EACH CHANNEL OMITTED
        CH7BGN                  ;BEGINNING IN CM OF CODE FOR CHANNEL 7
        CH7LST                  ;BEGINNING IN COMPUTER MEMORY OF LIST
CHNEND:
```

Program IV

MBD Code to Initialize Channels

```
MBDBGN: JVC                     ;JUMP VIA CTR TO SERVICE ROUTINE
        MVE+SPDR                ;POINTER TO LIST IS STORED IN PDR
        SHF+NRM+RIGHT+1         ;CHANGE POINTER FROM BYTE TO WORD POINTER
        DEP+DMAR+MRDR           ;READ FIRST WORD FROM LIST
        BCT+DCHBSY+4            ;WAIT FOR UNIBUS
        MVE+SMDR+DCTR           ;DEPOSIT FIRST WORD INTO CTR
        INM+SMAR+DMAR+MRDR      ;READ SECOND WORD FROM LIST
        BCT+DCHBSY+7            ;WAIT FOR UNIBUS
        MVE+SMDR                ;READ SECOND WORD INTO ATR
        SHF+NRM+RIGHT+1         ;CHANGE POINTER FROM BYTE TO WORD POINTER
        DEP+DILR                ;DEPOSIT SECOND WORD INTO ILR
        BCF+CTR15+15            ;CTR BIT 15 SETS PRL BY EXIT 1
        EXIT1                   ;EXIT 1 - SET PRL AND EXIT
        EXIT2                   ;EXIT 2 - RESET PRL, SET CEL, AND EXIT
```

Program V

MBD Code to Display Spectrum

```
CHØBGN=16                       ;BEGINNING OF CHANNEL Ø DISPLAY PROGRAM
.=MBDBGN+CHØBGN+CHØBGN          ;CPU PC IS BYTE POINTER
        MVE+SILR+DMAR+MRDR      ;READ CONSTANTS FROM CPU PROGRAM
        BCT+DCHBSY+CHØBGN+1     ;WAIT FOR UNIBUS
        DEM+SMDR+DDAR           ;DECREMENT FIRST WORD AND DEPOSIT IN DAR
        INM+SMAR+DMAR+MRDR      ;READ SECOND WORD
        BCT+DCHBSY+CHØBGN+4
        MVE+SMDR+DWCR           ;DEPOSIT SECOND WORD IN WCR
        INM+SMAR+DMAR+MRDR      ;READ THIRD WORD
        BCT+DCHBSY+CHØBGN+7
        MVE+SMDR+DCCR           ;DEPOSIT THIRD WORD IN CCR
        INM+SMAR+DMAR+MRDR      ;READ FOURTH WORD
        BCT+DCHBSY+CHØBGN+12
        LOD+SHFMSK              ;BITS Ø-3 CONTAIN NUMBER OF BITS SHIFTED
        AND+SMDR+DMDR           ;BIT 1Ø CONTAINS (RIGHT = Ø, LEFT = 1)
        LOD+SHFNRM              ;SHFNRM = SHIFT NORMAL INSTRUCTION
        ADT+SMDR                ;COMBINE INSTRUCTION AND NUMBER OF BITS
        STO+YSCALE              ;COMPLETE SCALE INSTRUCTION FOR Y DATA
        INM+SMAR+DMAR+MRDR      ;READ FIFTH WORD
        BCT+DCHBSY+CHØBGN+21
        MVE+SMDR+DGP2           ;LOW BYTE OF FIFTH WORD = X INCREMENT
        SHF+NRM+RIGHT+1Ø        ;HIGH BYTE OF FIFTH WORD = X OFFSET
        SHF+NRM+LEFT+2          ;DIVIDED BY 4
        DEP+DGP1                ;GP1 CONTAINS X OFFSET
        SHF+NRM+LEFT+6          ;REMOVE HIGH BYTE
        EOR+SGP2+DGP2           ;GP2 CONTAINS X INCREMENT
        LCI+DISPLY              ;POINT INTERRUPTS TO ITERATIVE DISPLAY
        EXIT1                   ;RELINQUISH CONTROL TO HIGHER PRIORITIES
DISPLY=CHØBGN+32                ;ITERATIVE DISPLAY OF SPECTRUM
        INM+SDAR+DMAR+MRDR      ;READ NEXT Y POINT
        MVE+SGP1+DBDR           ;LOAD X FROM GP1 INTO BDR FOR CAMAC
        MVE+SCCR+DBAR+BRN       ;EXECUTE CAMAC COMMAND TO LOAD X IN AØ
        BCT+DCHBSY+DISPLY+3     ;WAIT FOR UNIBUS
        MVE+SMDR                ;PUT NEW Y POINT INTO ATR
YSCALE=DISPLY+5                 ;SCALE FACTOR SHIFT INSTRUCTION FOR Y
        SHF+NRM                 ;SHF+NRM + BITS Ø-3 AND 1Ø FROM CPU
        BCT+BRBSY+DISPLY+6      ;WAIT FOR CAMAC BRANCH
        DEP+DBDR                ;LOAD Y INTO BDR FOR CAMAC
        LOD+Y                   ;LOAD Y AND INTENSIFY = CNAF + F18 + A1
        ADT+SCCR+DBAR+BRN       ;EXECUTE CAMAC COMMAND TO LOAD Y IN A1
        MVE+SGP2                ;X INCREMENT IS IN GP2
        ADT+SGP1+DGP1           ;INCREMENT X POSITION IN GP1
        DEM+SWCR                ;DECREMENT WORD COUNT
        BCT+BRBSY+DISPLY+15     ;WAIT FOR CAMAC BRANCH
        BCF+ZERO+DISPLY-1       ;DISPLAY NEXT POINT IF WCR NOT ZERO
```

Program V (continued)

```
        LCI+CHØBGN              ;BEGIN AGAIN ON NEXT SPECTRUM
        EXIT1                   ;RELINQUISH CONTROL TO HIGHER PRIORITIES
SHFMSK=DISPLY+21        2Ø17    ;BITS Ø-3 AND 1Ø FOR SHIFT MASK
SHFNRM=SHFMSK+1         SHF+NRM ;SHIFT NORMAL INSTRUCTION
Y=SHFNRM+1              2ØØØ1   ;CNAF = A1 + F2 (F18) FOR LOAD Y INTENSIFY
```

Program VI

MBD Code to Transfer Data from CAMAC to Computer Memory

```
CH7BGN=CHØBGN+1ØØ               ;BEGINNING OF CHANNEL 7 TRANSFER PROGRAM
.=MBDBGN+CH7BGN+CH7BGN          ;CPU PC IS BYTE POINTER
        DEM+SILR+DMAR+MRDH      ;POINTER TO BUFFER IS STORED IN ILR
        LCI+CH7LST              ;POINT   CM TO LIST OF CONSTANTS
        MVE+SCM+DWCR            ;RETRIEVE WORD COUNT FROM LIST
        INM+SCTR                ;POINT CM TO CAMAC CNAF COMMANDS
        BCT+DCHBSY+CH7BGN+4     ;WAIT FOR UNIBUS
        JVA+CH7RD+4             ;START TRANSFER WITH CAMAC COMMAND
LOOP7+CH7BGN+6                  ;TRANSFER LOOP
        BCT+DCHBSY+LOOP7        ;WAIT FOR UNIBUS
        INM+SMAR+DMAR           ;POINT TO NEXT LOCATION IN BUFFER
        BCT+BRBSY+LOOP7+2       ;WAIT FOR CAMAC BRANCH
        MVE+SBDR+DMDR+MWTH      ;WRITE WORD INTO COMPUTER MEMORY
        MVE+SCM+DBAR+BRN        ;READ NEXT WORD FROM CAMAC
        INM+SCTR                ;CNAF COMMANDS STORED IN CM LIST
        DEM+SWCR                ;CHECK FOR END OF WORD COUNT
        BCF+ZERO+LOOP7          ;CONTINUE WITH NEXT WORD
CH7SET=LOOP7+1Ø                 ;RESET PARAMETERS
        XEQINT                  ;INTERRUPT COMPUTER CPU FOR SERVICE
        INM+SMAR+SILR           ;STORE NEW BUFFER POINTER IN SILR
        LCI+CH7BGN              ;POINT INTERRUPT TO BEGINNING
        BCT+DCHBSY+CH7SET+3     ;WAIT FOR UNIBUS
        EXIT2                   ;EXIT AND RELEASE UNIBUS
CH7LST=CH7SET+5                 ;LIST OF WCR AND CAMAC CNAF COMMANDS
        25                      ;WORD COUNT (21 CNAF COMMANDS)
        1Ø2Ø                    ;FIRST CNAF COMMAND (FØ C1 N1 AØ)
        - - -                   ;REST OF LIST OF CNAF COMMANDS
MBDEND:
```

---

left in the ILR at the end of the transfer is the location of the next free address. An XEQ INT instruction at the end of the transfer alerts the computer CPU to the presence of a new block of data in the buffer of the computer memory.

The block transfer is effected in the loop of instructions between LOOP7 and CH7SET. Note that the write instruction holds the UNIBUS to minimize waiting time (the first read instruction is a dummy instruction to acquire and hold the UNIBUS). For most combinations of MBD and computer memory, the timing is such that the BCT instructions in this loop may be omitted so that the time for transfer is only 6 cycles of the MBD per word. If it is

inadvisable to use the hold made for data channel transfers (becasue of disk manipulations, etc.) the MWTH instruction can be replaced with a MWTR instruction, in which case the BCT+DCHBSY+LOOP7 instruction must be retained, and the MRDH instruction removed. The EXIT 2 instruction at the end releases the hold on the UNIBUS as well as relinquishing control to other channels until the next interrupt on channel 7.

Acknowledgements

I am deeply indebted to Kenneth A. Klare for an introduction to the language of CAMAC and for sample programs, and to Lavon R. Biswell and Robert E. Rajala for providing detailed information on the operation of the MBD processor. Thanks are due also to Randolph H. Jeppesen (U. of Montana), Anne C. Niethammer, and Richard F. Thomas for helpful comments.

References

1. J. A. Buchanan and H. V. Jones, "CAMAC Multi-Microprogrammed IO Processor," IEEE Trans. on Nucl. Sci., Vol. NS-19, No. 1, Feb. 1972, p. 682.

2. L. R. Biswell and R. E. Rajala, "A Microprogrammed Branch Driver (MBD) for a PDP-11 Computer," LA-4916-MS, Los Alamos Scientific Laboratory, Los Alamos, N.M. 87544, April 1972.