Datasheet for the **MICRO** Processor

B. Rasmus Anthin

April 26, 2000

1 General Description

The MICRO processor is a 8-bit processor with a 8-bit address bus. The CPU has a total of 138 instructions. The instructions has the following addressing modes:

- inherent
- immediate
- absolute
- relative

The *inherent* instructions have no following operands, all information about the operation comes with the instruction itself. Examples of inherent instructions are *INCA*, *CLRA*, *ROLA* and *RTS*. The *immediate* instructions have one operand which contents are often used as pure data for the operation to be performed. Examples of immediate instuctions are *LDA #Data*, *ADDA #Data* and *ORA #Data*. The *absolute* instructions have one operand which contains the address in which the operation should give it result to. Examples of absolute instructions are *STA Adr*, *SUBA Adr* and *ANDA Adr*. The *relative* instructions have one operand and acts somewhat similar to the absolute instructions, but instead the operand tells the offset from the value in the *PC-reg* in which address the result should be put in the memory or from which address data should be taken from the memory to be used by the operation.

The CPU contains mainly two parts who of which can be divided into several subparts:

control unit This is the "brain" of the CPU who controls the actions within the datapath and external units. It has two external signals which is used to interrupt the execution; it is the *RESET* signal and the *IRQ* signal. The RESET signal breaks the execution completely and starts it all over again. The IRQ signal interrupts the execution and jumps to another program and then jumps back when finished.

The countrol unit can be made in two ways:

static-coupled control unit. This is made by a state counter and a net of logical gates divided into different parts, each for every flag or controlsignal. This CPU uses this variant of control unit.

micro-programmable control unit. This is maintained with a micromemory containing all the CPU-instructions in separate areas. Each address corresponds to its instructions order-number. A micro-counter keeps track of the instructions executed within the micro-memory. This method is very rarely used today, but is useful when one wants to change the CPU's instruction-set.

data path This is the "body" of the microprocessor. The data path could be likened as a worker and the control unit as its boss who says what the data path should do with the data input. The data path contains following parts:

accumulators are referred to as registers. MICRO has two general registers, *A* and *B*. The registers are like containers for data. Alternatively, you could call them temporary variables.

ALU are the "hart" of the CPU. This part performs all the computations in the processor and therefore also in the computer! ALU is short for **A**rithmetic **L**ogic **U**nit.

CC-register is the **C**ondition **C**ode register. This register collects the flags from the ALU. The flags are used by the control unit to decide when a branch or a jump to another address should be performed. There are a total of five flags which are:

- **H**alf carry. This is the carry bit between the two nibbles in a byte.
- **N**egative. This flag indicates — when using 2nd complementary form — whether the result from the ALU is negative or not.
- **Z**ero. This flag indicates wheter the result from the ALU is zero or not.
- O**V**erflow. This flag indicates — when using 2nd complementary form — whether the result has wrong sign or not. E.g. if $127 + 40 = -167$ or if $-100 - 30 = 130$.
- **C**arry. This flag indicates carry from the most significant bit in the resulting data byte when addition or subtraction has been performed.

X-register is the index register which is frequently used when creating lists of variables in the memory.

S-register is a counter which keeps track of the address in the memory before the CPU performs a jump to another memory address.

PC is the **P**rogram **C**ounter. The PC keeps track of the current memory address containing the current instruction to be executed.

MA-register is the **M**emory **A**dress register which ports out the address to address bus (to the memory) from the data bus.

2 Instruction Set

This section deals with the instructions the MICRO processor has. They are more than enough to make complex programs such as controlling programs for external devices or games like snake for example.

2.1 Symbols

The symbols explained here are used throughout this section. The convetions are:

$number	number is in hex format.
%number	number is in binary format.
number	number is in decimal format.
OP	Operationcode for an instruction in hex.
#	Number of bytes an instruction uses. Is also used to tell that the operand is pure data.
∼	Number of clock cycles used to execute (and fetch) an instruction.
A	Data in register A.
A'	Data in register A is bitwise complemented.
$M(Adr)$	Data in memory at address Adr.
$M'(Adr)$	Data in memory at address Adr is bitwise complemented.
H	Halfcarry flag.
N	Sign flag.
Z	Zero flag.
V	Overflow flag.
C	Carry flag.
a	"Affected". Is used to show if a flag is affected by the operation.
•	"Not affected". The opposite to *a*.
0	Set to zero. Usually used to show that a flag is set to zero by the operation.
–	Undefined value. Used to show that a flag gets a random value from the operation.
CC	Condition Code. Contents in the flag-register, that is all flag-values.
n	Offset. Used in relative addressing modes.
EA	Effective Adress.
Offs	Offset from current adress.

2.2 No Operation

Operation type	*Name*	*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
No operation	NOP	00	1	3	–	•	•	•	•	•

2.3 Simple Data Transfer Operations

Operation type	*Name*		*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
Transfer	TFR	A,B	01	1	3	$A \rightarrow B$	•	•	•	•	•
	TFR	B,A	02	1	3	$B \rightarrow A$	•	•	•	•	•
	TFR	A,CC	03	1	3	$A \rightarrow CC$	a	a	a	a	a
	TFR	CC,A	04	1	3	$CC \rightarrow A$	•	•	•	•	•
	TFR	X,S	05	1	3	$X \rightarrow S$	•	•	•	•	•
	TFR	S,X	06	1	3	$S \rightarrow X$	•	•	•	•	•
Exchange	EXG	A,B	07	1	5	$A \leftrightarrow B$	•	•	•	•	•
	EXG	A,CC	08	1	5	$A \leftrightarrow CC$	a	a	a	a	a
	EXG	B,CC	09	1	5	$B \leftrightarrow CC$	a	a	a	a	a
	EXG	X,S	0A	1	5	$X \leftrightarrow S$	•	•	•	•	•
Load	LDA	Adr	0B	2	5	$M(Adr) \rightarrow A$	•	•	•	•	•
	LDB	Adr	0C	2	5	$M(Adr) \rightarrow B$	•	•	•	•	•
	LDX	Adr	0D	2	5	$M(Adr) \rightarrow X$	•	•	•	•	•
	LDS	Adr	0E	2	5	$M(Adr) \rightarrow S$	•	•	•	•	•
	LDA	#Data	0F	2	4	$Data \rightarrow A$	•	•	•	•	•
	LDB	#Data	10	2	4	$Data \rightarrow B$	•	•	•	•	•
	LDX	#Data	11	2	4	$Data \rightarrow X$	•	•	•	•	•
	LDS	#Data	12	2	4	$Data \rightarrow S$	•	•	•	•	•
Store	STA	Adr	13	2	5	$A \rightarrow M(Adr)$	•	•	•	•	•
	STB	Adr	14	2	5	$B \rightarrow M(Adr)$	•	•	•	•	•
	STX	Adr	15	2	5	$X \rightarrow M(Adr)$	•	•	•	•	•
	STS	Adr	16	2	5	$S \rightarrow M(Adr)$	•	•	•	•	•

2.4 Logic Operations

Operation type	*Name*		*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
AND	ANDA	Adr	17	2	7	$A \wedge M(Adr) \rightarrow A$	–	a	a	0	0
	ANDB	Adr	18	2	7	$B \wedge M(Adr) \rightarrow B$	–	a	a	0	0
	ANDA	#Data	19	2	6	$A \wedge Data \rightarrow A$	–	a	a	0	0
	ANDB	#Data	1A	2	6	$B \wedge Data \rightarrow B$	–	a	a	0	0
OR	ORA	Adr	1B	2	7	$A \vee M(Adr) \rightarrow A$	–	a	a	0	0
	ORB	Adr	1C	2	7	$B \vee M(Adr) \rightarrow B$	–	a	a	0	0
	ORA	#Data	1D	2	6	$A \vee Data \rightarrow A$	–	a	a	0	0
	ORB	#Data	1E	2	6	$B \vee Data \rightarrow B$	–	a	a	0	0
Exclusive-OR	EORA	Adr	1F	2	6	$A \oplus M(Adr) \rightarrow A$	–	a	a	0	0
	EORB	Adr	20	2	6	$B \oplus M(Adr) \rightarrow B$	–	a	a	0	0
	EORA	#Data	21	2	6	$A \oplus Data \rightarrow A$	–	a	a	0	0
	EORB	#Data	22	2	6	$B \oplus Data \rightarrow B$	–	a	a	0	0
Complement	COMA		23	1	4	$A' \rightarrow A$	–	a	a	0	–
	COMB		24	1	4	$B' \rightarrow B$	–	a	a	0	–
	COM	Adr	25	2	6	$M'(Adr) \rightarrow M(Adr)$	–	a	a	0	–
Flag manipulation	ANDCC	#Data	26	2	6	$CC \wedge Data \rightarrow CC$	a	a	a	a	a
	ORCC	#Data	27	2	6	$CC \vee Data \rightarrow CC$	a	a	a	a	a

2.5 Arithmetic Operations

Operation type	*Name*	*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
Add	ADDA Adr	28	2	7	$A + M(Adr) \rightarrow A$	a	a	a	a	a
	ADDB Adr	29	2	7	$B + M(Adr) \rightarrow B$	a	a	a	a	a
	ADDA #Data	2A	2	6	$A + Data \rightarrow A$	a	a	a	a	a
	ADDB #Data	2B	2	6	$B + Data \rightarrow B$	a	a	a	a	a
Subtract	SUBA Adr	2C	2	7	$A - M(Adr) \rightarrow A$	–	a	a	a	a
	SUBB Adr	2D	2	7	$B - M(Adr) \rightarrow B$	–	a	a	a	a
	SUBA #Data	2E	2	6	$A - Data \rightarrow A$	–	a	a	a	a
	SUBB #Data	2F	2	6	$B - Data \rightarrow B$	–	a	a	a	a
Negate (2's-compl)	NEGA	30	1	5	$A' + 1 \rightarrow A$	–	a	a	a	a
	NEGB	31	1	5	$B' + 1 \rightarrow B$	–	a	a	a	a
	NEG Adr	32	2	7	$M'(Adr) + 1 \rightarrow M(Adr)$	–	a	a	a	a
Arithmetic shift left	ASLA	33	1	4	$2 \cdot A \rightarrow A$	–	a	a	a	a
	ASLB	34	1	4	$2 \cdot B \rightarrow B$	–	a	a	a	a
	ASL Adr	35	2	6	$2 \cdot M(Adr) \rightarrow M(Adr)$	–	a	a	a	a
Rotate left	ROLA	36	1	4	$2 \cdot A + C \rightarrow A$	–	a	a	a	a
	ROLB	37	1	4	$2 \cdot B + C \rightarrow B$	–	a	a	a	a
	ROL Adr	38	2	6	$2 \cdot M(Adr) + C \rightarrow M(Adr)$	–	a	a	a	a
Increment	INCA	39	1	4	$A + 1 \rightarrow A$	–	a	a	a	a
	INCB	3A	1	4	$B + 1 \rightarrow B$	–	a	a	a	a
	INC Adr	3B	2	6	$M(Adr) + 1 \rightarrow M(Adr)$	–	a	a	a	a
Decrement	DECA	3C	1	4	$A - 1 \rightarrow A$	–	a	a	a	a
	DECB	3D	1	4	$B - 1 \rightarrow B$	–	a	a	a	a
	DEC Adr	3E	2	6	$M(Adr) - 1 \rightarrow M(Adr)$	–	a	a	a	a
Clear	CLRA	3F	1	4	$0 \rightarrow A$	0	0	1	–	0
	CLRB	40	1	4	$0 \rightarrow B$	0	0	1	–	0
	CLR Adr	41	2	5	$0 \rightarrow M(Adr)$	0	0	1	–	0

2.6 Test Operations

Operation type	*Name*	*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
Compare	CMPA Adr	42	2	6	$A - M(Adr)$	–	a	a	a	a
	CMPB Adr	43	2	6	$B - M(Adr)$	–	a	a	a	a
	CMPX Adr	44	2	6	$X - M(Adr)$	–	a	a	a	a
	CMPS Adr	45	2	6	$S - M(Adr)$	–	a	a	a	a
	CMPA #Data	46	2	5	$A - Data$	–	a	a	a	a
	CMPB #Data	47	2	5	$B - Data$	–	a	a	a	a
	CMPX #Data	48	2	5	$X - Data$	–	a	a	a	a
	CMPS #Data	49	2	5	$S - Data$	–	a	a	a	a
Zero or minus	TSTA	4A	1	3	$A - 0$	–	a	a	0	0
	TSTB	4B	1	3	$B - 0$	–	a	a	0	0
	TST Adr	4C	2	5	$M(Adr) - 0$	–	a	a	0	0
Bit test	BITA Adr	4D	2	6	$A \wedge M(Adr)$	–	a	a	0	0
	BITB Adr	4E	2	6	$B \wedge M(Adr)$	–	a	a	0	0
	BITA #Data	4F	2	5	$A \wedge Data$	–	a	a	0	0
	BITB #Data	50	2	5	$B \wedge Data$	–	a	a	0	0

2.7 Jump Operations

Operation type	*Name*		*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
Unconditional jump	JMP	Adr	51	2	4	$Adr \rightarrow PC$	•	•	•	•	•
Unconditional branch	BRA	Offs	52	2	6	$PC + Offs \rightarrow PC$	•	•	•	•	•
Conditional branch	BMI	Offs	53	2	6	$If\ N = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BPL	Offs	54	2	6	$If\ N = 0:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BEQ	Offs	55	2	6	$If\ Z = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BNE	Offs	56	2	6	$If\ Z = 0:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BVS	Offs	57	2	6	$If\ V = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BVC	Offs	58	2	6	$If\ V = 0:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BCS	Offs	59	2	6	$If\ C = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BCC	Offs	5A	2	6	$If\ C = 0:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BHI	Offs	5B	2	6	$If\ C' \cdot Z' = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BLS	Offs	5C	2	6	$If\ C + Z = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BGT	Offs	5D	2	6	$If\ (N \oplus V)' \cdot Z' = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BGE	Offs	5E	2	6	$If\ (N \oplus V)' = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BLE	Offs	5F	2	6	$If\ (N \oplus V) + Z = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
	BLT	Offs	60	2	6	$If\ (N \oplus V) = 1:$ $PC + Offs \rightarrow PC$	•	•	•	•	•
Jump to subroutine	JSR	Adr	61	2	7	$S - 1 \rightarrow S$ $PC \rightarrow M(S)$ $Adr \rightarrow PC$	•	•	•	•	•
Return from subroutine	RTS		62	1	4	$M(S) \rightarrow PC$ $S + 1 \rightarrow S$	•	•	•	•	•
Branch to subroutine	BSR	Offs	63	2	8	$S - 1 \rightarrow S$ $PC \rightarrow M(S)$ $PC + Offs \rightarrow PC$	•	•	•	•	•
Return from interrupt	RTI		64	1	12	$M(S) \rightarrow A$ $S + 1 \rightarrow S$ $M(S) \rightarrow B$ $S + 1 \rightarrow S$ $M(S) \rightarrow CC$ $S + 1 \rightarrow S$ $M(S) \rightarrow X$ $S + 1 \rightarrow S$ $M(S) \rightarrow PC$ $S + 1 \rightarrow S$	a	a	a	a	a

2.8 Pushing Data to Stack and Pulling Data from Stack

Operation type	*Name*	*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
Push accumulator A	PSHS A	65	1	5	$S-1 \rightarrow S$ $A \rightarrow M(S)$	•	•	•	•	•
Push accumulator B	PSHS B	66	1	5	$S-1 \rightarrow S$ $B \rightarrow M(S)$	•	•	•	•	•
Push register CC	PSHS CC	67	1	5	$S-1 \rightarrow S$ $CC \rightarrow M(S)$	•	•	•	•	•
Push register X	PSHS X	68	1	5	$S-1 \rightarrow S$ $X \rightarrow M(S)$	•	•	•	•	•
Pull accumulator A	PULS A	69	1	4	$M(S) \rightarrow A$ $S+1 \rightarrow S$	•	•	•	•	•
Pull accumulator B	PULS B	6A	1	4	$M(S) \rightarrow B$ $S+1 \rightarrow S$	•	•	•	•	•
Pull register CC	PULS CC	6B	1	4	$M(S) \rightarrow CC$ $S+1 \rightarrow S$	a	a	a	a	a
Pull register X	PULS X	6C	1	4	$M(S) \rightarrow X$ $S+1 \rightarrow S$	•	•	•	•	•

2.9 Increasing/Decreasing the X-register

Operation type	*Name*	*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
Load effective address	LEAX ,–X	6D	1	4	$X-1 \rightarrow X$	•	•	•	•	•
	LEAX ,X+	6E	1	4	$X+1 \rightarrow X$	•	•	•	•	•
	LEAX n,X	6F	2	6	$X+n \rightarrow X$	•	•	•	•	•
	LEAX A,X	70	1	5	$X+A \rightarrow X$	•	•	•	•	•
	LEAX B,X	71	1	5	$X+B \rightarrow X$	•	•	•	•	•

2.10 Data Transfer, Addressing via the X-register

Operation type	*Name*	*OP*	#	∼	*RTN*	*H*	*N*	*Z*	*V*	*C*
Load	LDA ,X	72	1	4	$M(X) \rightarrow A$	•	•	•	•	•
	LDB ,X	73	1	4	$M(X) \rightarrow B$	•	•	•	•	•
	LDA ,X+	74	1	5	$M(X) \rightarrow A$ $X+1 \rightarrow X$	•	•	•	•	•
	LDB ,X+	75	1	5	$M(X) \rightarrow B$ $X+1 \rightarrow X$	•	•	•	•	•
	LDA ,–X	76	1	5	$X-1 \rightarrow X$ $M(X) \rightarrow A$	•	•	•	•	•
	LDB ,–X	77	1	5	$X-1 \rightarrow X$ $M(X) \rightarrow B$	•	•	•	•	•
	LDA n,X	78	2	7	$M(n+X) \rightarrow A$	•	•	•	•	•
	LDB n,X	79	2	7	$M(n+X) \rightarrow B$	•	•	•	•	•
	LDA A,X	7A	1	6	$M(A+X) \rightarrow A$	•	•	•	•	•
	LDB A,X	7B	1	6	$M(A+X) \rightarrow B$	•	•	•	•	•
	LDA B,X	7C	1	6	$M(B+X) \rightarrow A$	•	•	•	•	•
	LDB B,X	7D	1	6	$M(B+X) \rightarrow B$	•	•	•	•	•
Store	STA ,X	7E	1	4	$A \rightarrow M(X)$	•	•	•	•	•
	STB ,X	7F	1	4	$B \rightarrow M(X)$	•	•	•	•	•
	STA ,X+	80	1	5	$A \rightarrow M(X)$ $X+1 \rightarrow X$	•	•	•	•	•
	STB ,X+	81	1	5	$B \rightarrow M(X)$ $X+1 \rightarrow X$	•	•	•	•	•
	STA ,–X	82	1	5	$X-1 \rightarrow X$ $A \rightarrow M(X)$	•	•	•	•	•
	STB ,–X	83	1	5	$X-1 \rightarrow X$ $B \rightarrow M(X)$	•	•	•	•	•
	STA n,X	84	2	7	$A \rightarrow M(n+X)$	•	•	•	•	•
	STB n,X	85	2	7	$B \rightarrow M(n+X)$	•	•	•	•	•
	STA A,X	86	1	6	$A \rightarrow M(A+X)$	•	•	•	•	•
	STB A,X	87	1	6	$B \rightarrow M(A+X)$	•	•	•	•	•
	STA B,X	88	1	6	$A \rightarrow M(B+X)$	•	•	•	•	•
	STB B,X	89	1	6	$A \rightarrow M(B+X)$	•	•	•	•	•