J. C. Cocke

BASIC COMPUTER MEMO #5

SUBJECT: A Checking Scheme for Data Flow

BY: F. B. Hartman

DATE: October 10, 1957

The flow diagram for the basic computer has been developed into the form shown in Figure 1. In order to realize the speeds required of this machine, it has been necessary to overlap operations by use of a 'pipeline'. By a 'pipeline' is meant a scheme of control whereby several stages of logic all perform simultaneously but on successive bytes. For example, for a five-stage 'pipeline', to process the bytes a, b, c, and d successively through stages I, II, III, IV, and V, a pattern of overlapping of operations is developed as shown below:

STAGES / CYCLES	I	II	III	IV	V
1	a	-	-	-	-
2	b	a	-	-	-
3	c	b	a	-	-
4	d	c	b	a	-
5	-	d	c	b	a
6	-	-	d	c	b
7	-	-	-	d	c
8	-	-	-	-	d

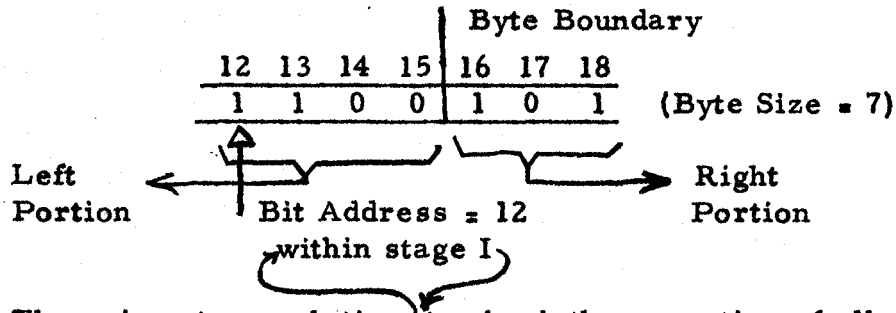
A checking scheme has been developed for the data flow shown in Figure 1, under the assumption that registers A, B, C, and D are provided with a parity bit for each 8 bits of information from Memory. The object of the checking scheme is (1) to continually update the parity bits as new information enters one of these registers, a byte at a time, from the Logical Unit, and (2) to detect failure of any component within the data flow path.

The checking scheme involves the following operations:

1. The Out Switch of Register AB selects 16 information bits and two (2) parity bits from the 128 information bits and 16 parity bits. Thus the performance of this Switch is checked by any subsequent parity check performed upon the data which passes through it. The parity bits are labeled P_j, P_k, .

2. The Byte Selector Left and Byte Selector Right, together with the Byte Selector OR, select eight (8) information bits from the 16 presented by the AB Out Switch. Parity bits P_l and P_r are generated for these selected bytes. Note that the sum of these two parity bits (modulo 2) is the required parity bit for the selected 8-bit byte which (ultimately) goes to the Logical Unit.

3. The Complementary Byte Selector Left and the Complementary Byte Selector Right select the left and right halves of the complementary byte (the eight (8)) bits that are <u>not</u> selected to go to the Logical Unit). Left and right refer to the two parts of the byte as divided by a byte boundary in register AB. The division into left and right selected or complementary bytes is determined by the bit address. Byte Boundaries occur between addresses 7 and 8, 15 and 16, ..., 127, and 0. The example which follows illustrates this process: (Note that parity bits P_{cl} and P_{cr} are generated for the bits coming out of the complementary byte selectors).



4. There is not enough time to check the operation of all the above circuitry, so the parity bits together with the data selected to go to the Logical Unit are stored in Register 1 or 1', depending upon the cycle. Since the Parity bits are carried along, all of the circuitry in Stage I can be checked by suitable parity checks performed later.

5. The Parity check upon operations in Stage I are performed in Stage II. The sum (modulo 2) of parity bits P_j, P_{cl}, and P_l is generated and if this is up, an error is signaled. A similar operation is performed for parity bits P_k, P_{cr}, and P_r to signal an error. The sum (modulo 2) of the Parity bits P_l and P_r is generated

and the out of phase output is taken as the parity bit of the selected byte. Parity bits are generated for the outputs of the Pass Gate and Complementary Pass Gate. Call them P_g and P_{cg}, respectively. The sum (modulo 2) of P_l, P_r, P_g, and P_{cg} is generated and if this is up, an error is signaled. In this way, the operation of the Pass Gate and the parity generator attached thereto, is adequately checked.
(Note: P_l and P_r must be updated as the data passes through the pass gate).

6. Stage III contains the Adder Unit and a Parity Bit Predictor. The operation of the Adder Unit is not checked by parity until stage IV, however.

7. In Stage IV, the Parity bit associated with the data undergoes modification in accordance with the requirements of decimal correction. The decimal correction is not checked until stage V, however.

8. In Stage V, the operation of Stage IV is checked and the result byte entered into the appropriate register together with the corresponding updated parity bits.

The operations that take place within Stage V deserve further consideration. First of all, the parity bits P_j, P_k, P_l, and P_r, which were generated in Stage I, are carried down the pipeline to Stage V. P_l and P_r are modified in Stage II in accordance with the Pass Gate operation. They are used to correctly update all the parity bits. In Figure 2 is illustrated an example showing how the Parity bits are revised. The Operand field lies in Register AB at address 21, field length is 14 and byte size is 4. The result is to be returned to Register AB. The operation is that of inverting all bits of the field.

To correctly update the parity bits, it is necessary to know two pieces of information:

1. Does the left (right) parity bit need to be altered?

2. What is the current status of the left (right) parity bit?

By calculating the left and right parity bits for the result byte, calling them P'_l and P'_r, respectively, it is possible to tell whether the corresponding parity bits of register AB must be changed. In calculating these parity bits,

the following rules are used:

1. If P_l and P'_l disagree, change the left parity bit (left relative to current bit address).

2. If the right parity bit (right relative to current bit address) needs to be altered and P_r disagrees with P'_r, then change the right parity bit.

A trigger is provided within Stage V to remember the current status of the left parity bit. Thus for the first byte the left parities disagree, P_j says that current left parity bit is on so P_c is turned off and the trigger is turned off. The right parities also disagree, so P_d, which was off, as indicated by P_k, is turned on.

For the second byte, the left parities agree and the right parity bit is unaffected, so all parity bits remain unaltered.

For the third byte, the machine has just passed a byte boundary, so the reference status of the left bit is now contained in P_j. The left parities disagree, so P_b is set to one and the trigger is set to one. The right parities disagree so P_c is set to one, since its previous status was zero, as indicated by the previous status of the trigger.

For the last byte, which is two bits long, the left parities agree and the right parity bit is unaffected, so all parity bits are unaltered.

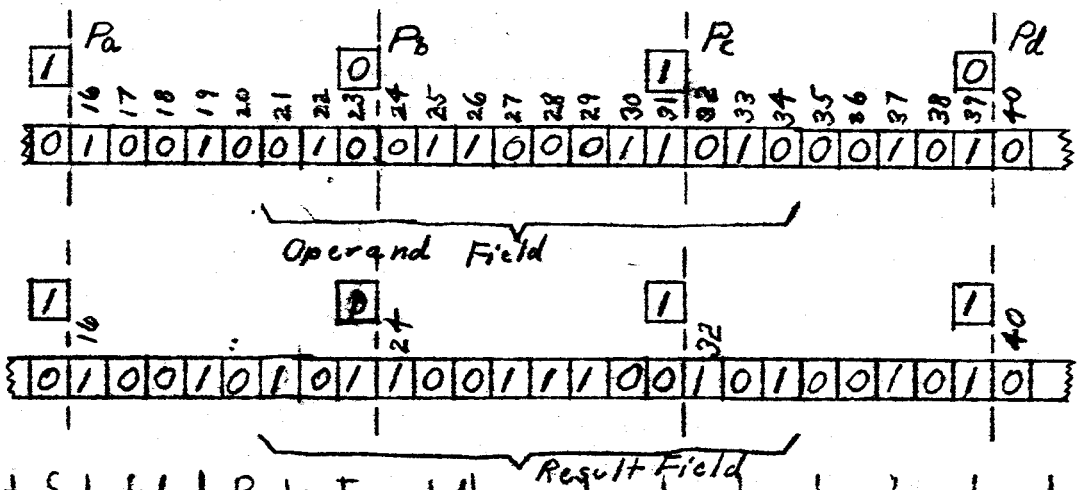
SUMMARY AND CONCLUSIONS

The checking scheme described above appears to permit a high degree of continuity of checking, in which any single-bit failure in the entire data flow path, including intermediate registers can be detected. By this scheme, the parity bits of the main registers A, B, C, and D are continually updated, so that the register always (for correct operation) contains the proper parity bits. It should be noted that the operations of Stage V cannot be checked in Stage V but must be done at a later stage, say as the data is sent to memory or sent again to the Logical Unit.

Frank B. Hartman

Frank B. Hartman

FBH/emb



Original Field-Register AB.

Result Field-Register AB

Address	Byte Selected from Reg AB	Byte Inserted into Reg AB	T	P_b	P_c	P_ℓ	P_r	P_ℓ'	P_r'	—	P_a	P_b	P_c	P_d	Notes
31	–	–	0	–	–	–	–	–	–		1	0	1	0	Initial Conditions
31	1/0/1 0 0 1 0	0/101----	0	P_c=1	P_d=0	0	0	1	1		1	0	0	1	
27	0 0 0 1 1 0 1 0	1 1 1 0 -/---	0	P_c=1	P_d=0	0	0	0	–		1	0	0	1	
23	0/0 1 1 0 0 0 1	1/100----	1	P_b=0	P_c=1	1	0	0	0		1	1	1	1	Machine has just passed byte boundary
21	0 1 0 0 1 1 0 0	10-/-----	1	P_b=0	P_c=1	0	1	0	–		1	1	1	1	Last Byte.

FIGURE 2. EXAMPLE SHOWING REVISION OF PARITY BITS

(Example consists of the operation of "inverting all bits" of a 14-bit field with address 21 and Byte Size of 4).

F.B. 10/8/57

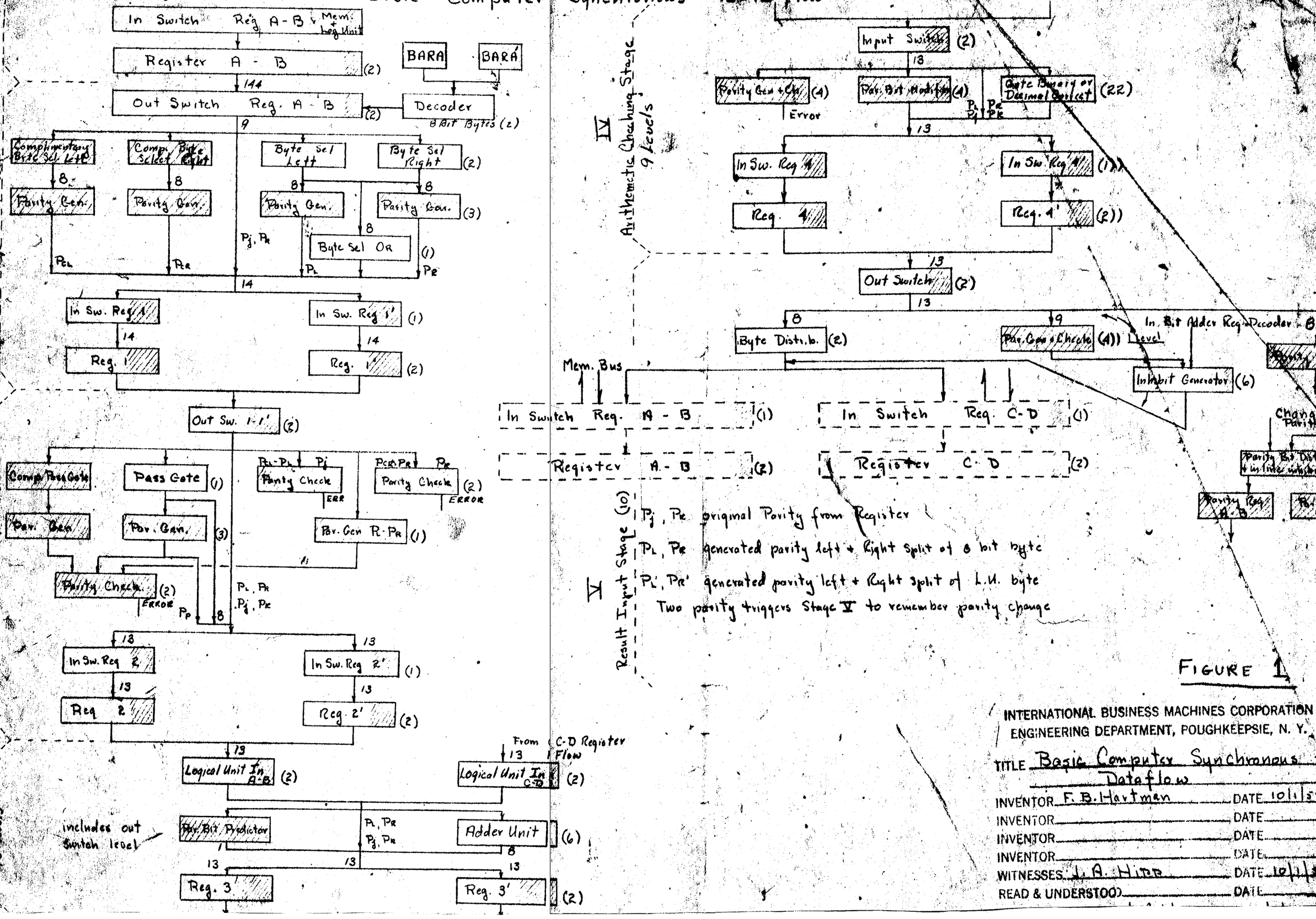


FIGURE 1

INTERNATIONAL BUSINESS MACHINES CORPORATION
ENGINEERING DEPARTMENT, POUGHKEEPSIE, N. Y.

TITLE Basic Computer Synchronous Dataflow

INVENTOR F. B. Hartman DATE 10/1/5

INVENTOR DATE

INVENTOR DATE

INVENTOR DATE

WITNESSES J. A. Hibb DATE 10/1/5

READ & UNDERSTOOD DATE