Local Memory and Logic Arrangement for Ultra-Low Power Array Processors

Ari Paasio
Technology Research Center (TRC)
University of Turku
Turku, Finland
ari.paasio@utu.fi

***Abstract*—Array processors, and vision chips in particular, have mostly been designed from maximum processing speed point of view. There are applications in e.g. surveillance field, where the image content is analyzed rather rarely and where on the other hand the power consumption is of greater importance due to battery operation functionality. In sensing applications it is customary to use a coarser sensing for triggering a finer tuned sensor, where the coarse sensor is optimized for lower power and where the sensing abilities have been relaxed to that of a triggering ability. In this paper we continue on reporting on the progress of PMOS only based cellular array processor by introducing a possible arrangement for local binary memory and local binary logic.**

***Keywords*— wave computing; local binary memory; local binary logic; ultra low-power; PMOS logic**

I. Introduction

Current state-of-the-art vision chips based on array processing can achieve processing speeds for analyzing tens of thousands of frames per second [1, 2] while consuming some hundreds of milliwatts power. The historical trend in the design of these types of processor arrays has been to pursuit extremely high frame rates with less attention paid to the power consumption. There are applications, where the frame rate is not required to be extremely high, but where the battery and/or harvester based operation and pursuit for long operating time set different design targets. Such applications can be found in e.g. surveillance arena, where even a lower resolution vision chip could act as a triggering means for higher resolution image analysis when something interesting is taking place in the monitored scene. In these cases, the targeted frame rates can be relatively low, in the order of one frame per few seconds. A new logic structure operating with mainly leakage currents of PMOS transistors was introduced in [3, 4] where also a 1-D CNN type propagation network circuitry was given with simulation results. Currently, the propagation network has been measured to operate correctly, as simulated, between the supply voltage range of 135mV to 200mV. The measurement results will be published separately, but the verified operability of the network has motivated the continuing work with the logic family in connection with array processor design. In this paper, we introduce means to incorporate local binary memories and local binary logic in connection with the propagation network.

II. PMOS logic multiplexer

The structures presented in this paper are built around a PMOS logic multiplexer, which will be introduced first. Later, the block functionalities are explained with the help of the operation of the multiplexer.

A. Conventional multiplexing

The schematics of the proposed multiplexer is shown in Fig. 1, where there are two inputs, A and B, to the multiplexer associated with the transistors M2 and M4, respectively. The control signals C1 and C2, associated with transistors M3 and

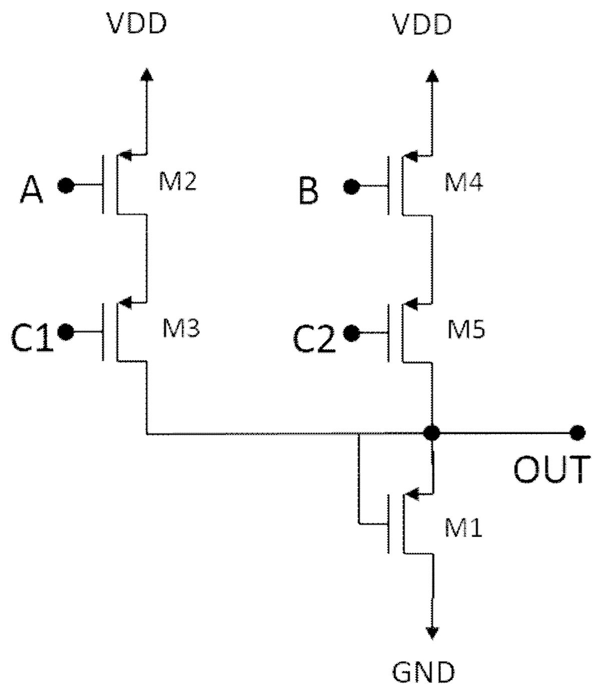

Fig. 1 Two-input multiplexer.

M5, are in basic operation mode assumed to be arranged in such a manner, that one of these signals is HIGH and the other one is LOW. When C1 is LOW, the inverted logic level of the signal A is present at the output. Similarly, when C2 is LOW, the inverted level of the signal B is at the output. Note, that in this typical case, one would need only one control signal, where the other one would be obtained by an inverter. However, due to the functionalities presented later, we introduce the multiplexer with two distinctively separate control signals.

B. *Logic within multiplexer*

There are (at least two) additional functionalities that can be obtained with the multiplexer of Fig. 1 that can be utilized in the local memory and logic arrangement. Namely, when both control signals are HIGH, the multiplexer outputs a logic level LOW. By utilizing this phenomenon, one does not require an additional global pull-up or pull-down functionality. The second feature of the PMOS logic multiplexer is that by having both control signals LOW, the output is a logical NAND of the inputs A and B and therefore, local logic can be performed easily without dedicated separate logic blocks.

III. PMOS LOGIC LATCH

The local memory element proposed for in-pixel binary storage is a 7-transistor latch shown in Fig. 2. Within the latch there are the multiplexer of Fig. 1 together with a PMOS logic inverter, where the non-inverted output of the latch is formed to the output of the inverter and where this output serves as a second input to the multiplexer. Note that an inverted version of the latch content is available at the input of the inverter. The read-out arrangement from the memory is explained later in accordance with higher level description and here we'll concentrate on the writing procedure as well as to the data storage mode.

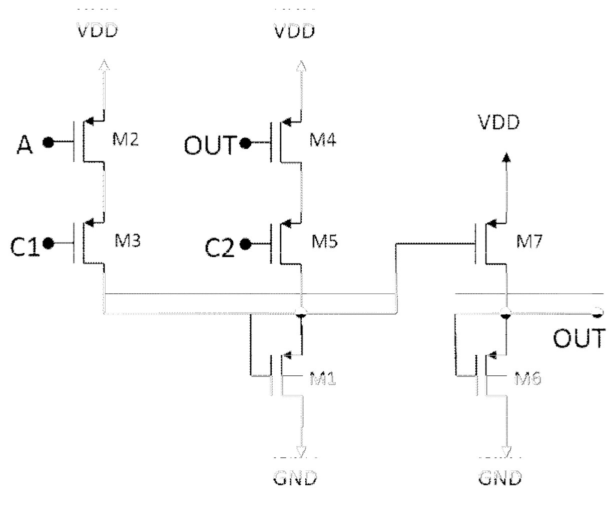

Fig. 2 Multiplexer based latch.

The latch can be written by setting the control signal C1 LOW and the control signal C2 high. Then, the output of the multiplexer will be the inverted version of the latch input A, whereas the output of the latch, node OUT, will be a copied logic level from A. With these control signal polarities, the latch will be transparent from input to output. The data is stored and isolated from the input A, i.e. the latch is opaque, by setting the signal C1 HIGH and the signal C2 LOW. With this arrangement, there effectively exists a cross connected pair of inverters similarly to a normal CMOS logic and the data is stored in this latch. Since the changes in internal signal levels are not fast, the relative timing between the control signals C1 and C2 going LOW and HIGH is not required to be highly controlled. Instead, more or less, non-overlapping signals can be used in the write control of the latch.

The latch can be easily SET to output level HIGH, by having both control signals being HIGH for a long enough time so that the multiplexer output has time to go LOW thus setting the output of the latch HIGH, after which the signal C2 can be set LOW normally. Moreover, by having both control signals LOW simultaneously, the latch effectively performs an AND operation between the stored level within the latch and the input A of the latch as described earlier in connection with the description of the multiplexer. If these more versatile operations, the SET and AND are to be implemented, the control signals C1 and C2 are preferably operated separately from each other instead of using complementary signals.

IV. PROCESSING ELEMENT LEVEL MEORY AND LOGIC ARRANGEMENT

The system level arrangement of various memories being able to read and write to each other as well as contribute and receive information from the propagation network is shown in Fig. 3. There are three latches of Fig. 2 incorporated with their outputs directly connected to a 4-input multiplexer. The multiplexer is similar to the one shown in Fig. 1 with now four separate inputs A, B, C and D and their associated control signals C1 to C4, respectively. By using a multiplexer at the outputs of the latches, the reading out of the contents of the latches is enabled. Moreover, since the output node of the propagation network is similarly non-buffered (see [4] for the propagation network schematics), the multiplexer arrangement works similarly to reading out the propagation network output.

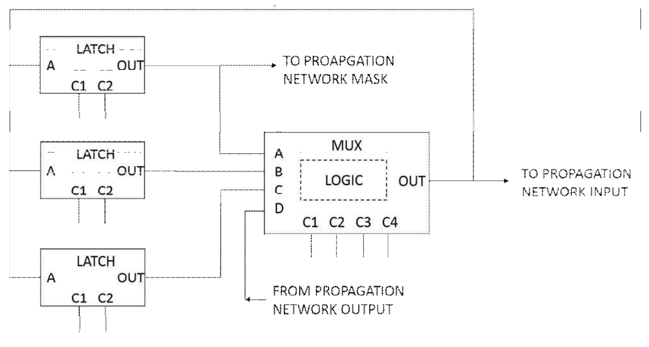
LATCH
A
OUT
C1 C2
TO PROAPGATION NETWORK MASK
LATCH
A
OUT
C1 C2
MUX
A
B
C
D
LOGIC
OUT
C1 C2 C3 C4
TO PROPAGATION NETWORK INPUT
LATCH
A
OUT
C1 C2
FROM PROPAGATION NETWORK OUTPUT

Fig. 3 Memory/local logic arrangement around main multiplexer.

The multiplexer output is connected to the inputs of the local memory latches so that each latch can read the output of the multiplexer either separately or simultaneously. Moreover, the output of the multiplexer can be used to control the initial condition of the propagation network directly. We suggest to allocate one latch to directly control the propagation network mask transistor, because the latches can easily be SET, where the HIGH output of the latch results in the mask being off. However, other arrangements for the mask, as well as providing the initial condition, are also possible.

As pointed out earlier, the multiplexer can also be used to perform a NAND operation between it's inputs. This is also possible with the four input multiplexer forming the core of the memory IO-system. However, if one requires e.g. a faster NOR or XOR execution, more functionality can be added directly to the multiplexer. A separate NOR/XOR gate is shown in Fig. 4, where the NOR operation is performed by transistors M7 and M8 in series and where this evaluation is executed by setting the signal NOR LOW. Similarly, the XOR signal activates and deactivates the pull-up path of the XOR evaluation tree formed by transistors M2 to M5. The XOR evaluation requires, in addition to two non-inverted signals A and B, also their inverted counterparts XA and XB. These inverted latch outputs are available at each individual input of the inverter within the latch and can be wired directly to the associated XOR-tree transistors. A typical arrangement would be to hard-wire two latches to the XOR-transistors and let the programmer or compiler to keep these particular latches reserved for local logic, when necessary. It is noted, that there needs not to be a separate pull-down transistor for the NOR/XOR inclusion directly to the main multiplexer, but that the pull-up network can directly be connected to the output of the multiplexer with shared pull-down arrangement.

V. SIMULATIONS

The simulations were performed with 130nm CMOS technology using Eldo simulator. In all the simulations, the supply voltage is 200mV and the bulk is also at 200mV. Each simulation is shown with nominal parameters in order to keep the visualization clearer. The simulated arrangement included two latches of Fig. 2 together with a three input multiplexer connected to the latches similarly to the arrangement in Fig. 3. In all the voltage waveform simulations, there are various control signals shown on the topmost boxes with C1, C2 and C3 on the top box representing the three control signals of the multiplexer. The input associated with C3 is expected to come from an outside source, like e.g. the propagation network. All the control signals are buffered by two PMOS inverters in series with these inverters having the same sizes as those within the latches. By this buffering, we want to demonstrate, that even rather slow control signals work properly. The latch control signals occupy the next two top boxes with S1 and XS1 associated with the first latch (related to C1) and S2 and XS2 associated with the second latch. The second box from the bottom shows the output of the main multiplexer OUTM, and the bottom box shows the outputs of the two latches denoted as OUT1 and OUT2. The time span of each simulation is 1ms.

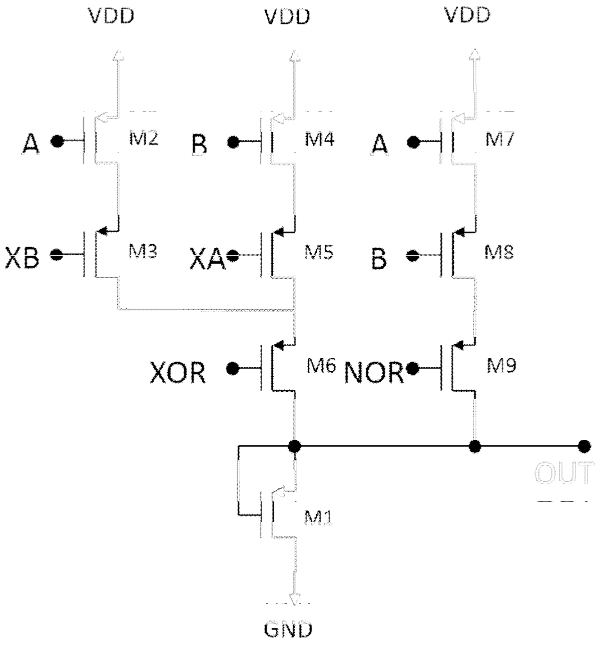

Fig. 4 Additional local logic arrangement for NOR and XOR.

In the first simulation, shown in Fig. 5, all the multiplexer control signals are initially HIGH yielding the corresponding output node to be LOW. Also, all the control signals associated with the latches are HIGH in the beginning resulting in a SET operation where the outputs of the latches are HIGH. At time instant 0.2ms XS1 and XS2 turn LOW and the HIGH values of both latches are kept. Next, at 0.4ms, the control signals of latch one are set to read mode, S1 turning LOW and XS1 turning HIGH. At this instant, the output of the MUX, still being LOW, is read to the latch one and this results in the OUT1 turning LOW. The latch one is restored to it's opaque mode at 0.6ms. Finally, at 0.8ms, the main multiplexer signal C1 is set active, i.e. LOW, and the multiplexer output adapts to the output of latch one, now giving the inverted version of the output of latch one.

The second simulation, shown in Fig. 6, starts similarly to the first simulation with all the control signals HIGH. At 0.2ms, the latch signals S1 and S2 are set to LOW resulting in write operation to the memories. Both outputs turn LOW, since the multiplexer output is LOW.

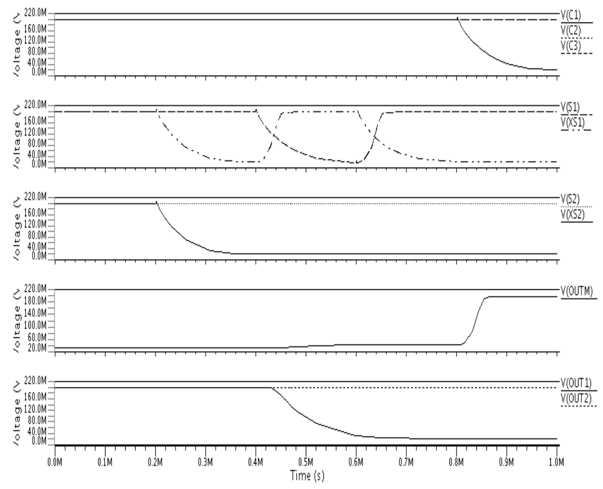

Fig. 5 Transient simulation with latch SET, read and write.

At 0.4ms, the control S2 is turned HIGH, but without turning the XS2 LOW. This will result in SET operation for the latch two and the output goes HIGH again. There is a roughly 40µs delay from the HIGH turning S2 signal at 100mV to that of the OUT2 reaching level 100mV giving some measure on the average speed of the latch. At 0.6ms, the signals S1 and XS1 change polarities resulting in correctly restoring the written LOW input level. Again, at 0.8ms, the multiplexer control signals are altered and in this simulation, both C1 and C2 go LOW effectively resulting in NAND operation of the latch outputs. As a result, since OUT1 is LOW, the multiplexer output turns HIGH.

The third simulation, in Fig. 7, shows the AND functionality in writing to the latches. The simulation starts similarly to the previous ones until 0.2ms, when signals XS1 and XS2 are set LOW. At this time instant, the latches are in

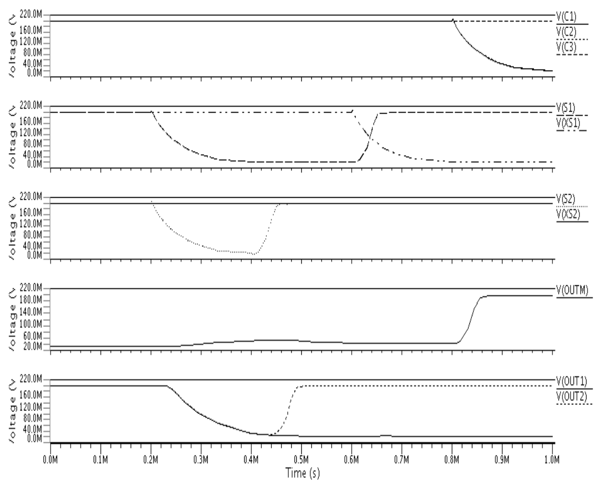

Fig. 6 Transient simulation with latches written, latch two SET, latch one memorized, and main multiplexer output NANDed.

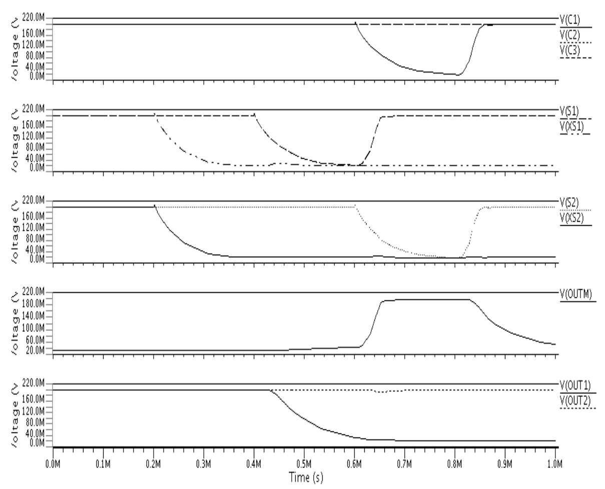

Fig. 7 Transient simulation with latches ANDed one by one and multiplexer RESET.

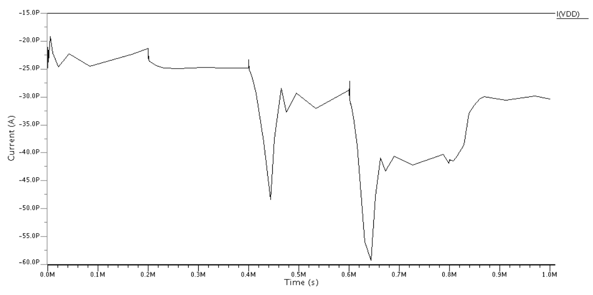

Fig. 8 Power supply current associated with simulation of Fig. 7.

the normal mode of keeping the data. At 0.4ms, the latch one is written to by setting the S1 LOW. However, the XS1 stays LOW and there is the AND operation performed within the latch. Since the input to that particular latch is LOW, the output turns from HIGH to LOW. At 0.6ms, the latch one is set to it's normal opaque mode, while the main multiplexer is set to read out the output of the latch one. This makes the OUTM turn HIGH and at the same time instant, the latch two is set to write mode in the AND type fashion by setting S2 LOW. Now, because both, the latch two output as well as the input are HIGH, the latch keeps it's HIGH output according to the AND operation. Finally, at 0.8ms, the latch two is restored to the normal opaque mode, while the multiplexer is RESET by turning C1 HIGH. The RESET of the multiplexer yields the corresponding output to go LOW.

Finally, we show the simulated power supply current profile in Fig. 8 for the two latches and the multiplexer, where the simulated performance was that shown in Fig. 7. The inverters used for buffering the control signals used a separate power supply and they are not included in Fig. 8.

VI. CONCLUSIONS

We have extended our PMOS logic based processing element functionality to include local memories and local logic. After the introduction of these building blocks, all the necessary elements have been proposed for implementing the basic binary processing tasks of an array processor. The multiplexer based memory and logic arrangement have been shown to offer versatile operation modes additionally to the basic IO modes typically present in local memories.

REFERENCES

[1] A. Lopich, and P. Dudek, "A SIMD Cellular Processor Array Vision Chip With Asynchronous Processing Capabilities," IEEE Trans. on Circuits and Systems – I, vol. 58, pp. 2420-2431, October 2011.

[2] M. Laiho, J. Poikonen, and A. Paasio, "MIPA4k: Mixed-Mode Cellular Processor Array", in "Focal-Plane Sensor-Processor Chips" A.Zarandy (Ed.), ISBN: 978-1-4419-6474-8, Springer, 2011.

[3] A. Paasio, "Ultra Low-Power Array Processor Propagation Circuit Arrangement," IEEE International Symposium on Circuits & Systems (ISCAS), Montreal, Canada, 2016.

[4] A. Paasio, "Ultra Low-Power Array Processor 1-D Test Structure Implementation,"The 15th International Workshop on Cellular Nanoscale Networks and their Applications (CNNA), Dresden, Germany, 2016.