Reduced complexity two-phase micropipeline latch controller

George S. Taylor and Gerard M. Blair
Department of Electrical Engineering,
The University of Edinburgh,
Edinburgh, Scotland
{gst,gerard}@ee.ed.ac.uk

Abstract

A latch control circuit for two-phase micropipelines is presented and compared to previously published alternatives. The circuit avoids the complex toggle element. It is smaller, faster and consumes less power than other two-phase implementations and compares favourably with four-phase latch controllers.

I Introduction

The asynchronous counterpart of the synchronous pipeline is the micropipeline [1]; the canonical form is a FIFO as shown in Fig. 1. Each stage in the pipeline can request that the next stage accept new data and acknowledge receipt of data from the previous stage, typically by using the bundled data interface protocol. Two forms of this protocol exist: two-phase, in which rising and falling edges are equivalent, and four-phase, in which falling edges are redundant. Fig. 2 illustrates the two-phase protocol.

Sutherland [1] describes a capture-pass latch suitable for use with the two-phase protocol. However, for the design of wide data-paths it is desirable to use a more compact latch with the complexity moved to a single controller circuit which operates a multi-bit data latch and controls the handshake signals to adjacent stages. Previous development of latch controllers for both two-phase and four-phase protocols can be found in [2] and [3].

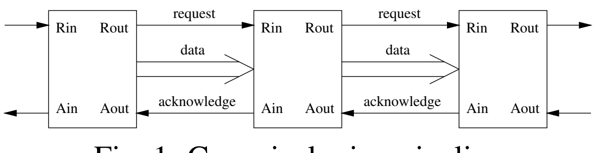

Fig. 1: Canonical micropipeline

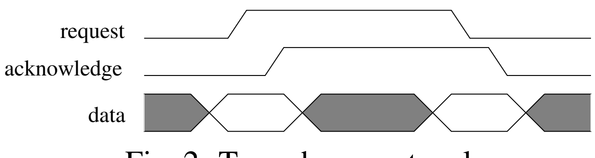

Fig. 2: Two-phase protocol

The two-phase latch controller [1, 2] shown in Fig. 3 is effectively a two-phase to four-phase converter (the XOR), followed by a four-phase to two-phase converter (the toggle). The toggle senses when the latch enable `En` has changed; typically the actual layout may route `En` through the data latch and then to the toggle. The data latch is transparent when `En` is high. In [2] this circuit is shown to be unfavourable when compared to a purely four-phase system.

Yun *et al*. [4] describe a two-phase micropipeline which avoids the conversion circuitry, leaving only the C-element and buffer, by using double-edge triggered flip-flops. The circuit has a much shorter cycle time than the semi-decoupled four-phase circuit [3] which has the shortest cycle time of the four-phase circuits. However, this is at the expense of substantially increased power dissipation and area due to the complex flip-flop circuit required per data bit. The speedup is also partly due to additional timing assumptions. In this paper we present a two-phase circuit which does not require the toggle element and which employs a conventional data latch.

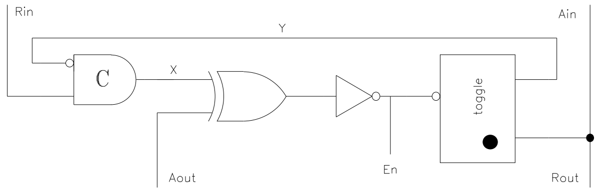

Fig. 3: Controller with toggle

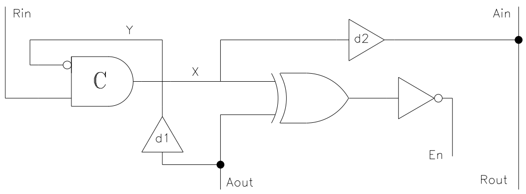

Fig. 4: With delay matching

II Latch controller without toggle

For the circuit in Fig. 3, an event on `Rin` causes an event on `Ain` and `Rout` once the data latch has captured, and an event on `Aout` causes an event on the feedback path `Y` once the data latch has returned to transparency mode. Therefore `Ain` and `Rout` can be generated by delaying `Rin` and `Y` can be generated by delaying `Aout`. This is shown in Fig. 4. Delay `d1` ensures all data latch bits are transparent for the minimum time required to capture new data, and `d2` ensures an acknowledgement is not sent to the previous stage before the data has been captured. Delays `d1` and `d2` may be obtained by careful simulation.

Separate delays for `Ain` and `Rout` may be used. For example, a shorter delay might be used to generate `Rout` than for `Ain`, if the data latch passes the data an appreciable time before the data is captured. If `Rout` is connected to the input of `d2` in Fig. 4, then the circuit is the equivalent of the 'fast-forward' circuit described in [2] where the C-element delay is not shorter than the data propagation time of the transparent data latch.

Both `d1` and `d2` can be removed to give a similar micropipeline to that used by Yun *et al* [4], in that both `Rout` and `Ain` are generated early. Therefore similar timing assumptions to those made by [4] would be required, with the additional constraint that the data latch must be transparent for a minimum time between captures.

Although the circuit of Fig. 4 can be implemented, careful simulation to find the correct delays is required and, if the load on `En` is large, a long chain of inverters may be required to produce each delay. Instead we propose the scheme described below.

III Latch controller without toggle or delay matching

The purpose of `d1` is to delay priming of the C-element to delay further requests until `En` is high. Likewise the purpose of `d2` is to delay generation of `Rout` and `Ain` until `En` is low. The circuit shown in Fig. 5[1] achieves this behaviour by using level-sensitive latches `l1` and `l2` to block or pass events as is shown in Fig. 5. The latch acts as an AND between a transition sensitive signal and a level sensitive signal.

[1] We thank Mark Josephs for pointing out subsequently that this circuit has been previously discussed at workshops by himself [5] and by IW Jones [6], and for forwarding to us copies of the viewfoils of these presentations.

It assumed that l1 and l2 operate on En at the same time, i.e. the fork leading to both is isochronic. If this is not the case then the possibility exists for l1 and l2 to be transparent at the same time allowing events to pass directly from Rin to Rout and Ain. Case 1: If, when En goes low, l1 is late in capturing, the Ain transmitted by l2 could result in a further Rin whilst l1 is still transparent, if Ain and Rin are connected through little or no circuitry. However, there is an acceptable safety margin here as the path normally involves the Aout to Rout delay of the previous stage. Case 2: If, when En goes high, l2 is late in capturing, a waiting Rin may pass straight through the transparent l1 and the still transparent l2. This race hazard exists solely inside the pipeline stage and does not involve an external path and is similar to one encountered in many synchronous designs. Additionally it is assumed that the weak inverter feedback circuits present in the l1 and l2 elements operate before further input changes to the elements occur.

The actual implementation is shown in Fig. 6. Inverter b1 provides buffering to drive the multi-bit data latch. Complementary signals are used to drive pass-transistor latches l1 and l2, p4 is required to ensure both signals are subject to the data latch capture/pass delay. For the fast-forward version inverter p2 is required to obtain the correct polarity for Rout-ff. Since p2 is required anyway, an efficient complementary pass-transistor XOR gate is used requiring p1. To counter the potential race hazard (case 2 above) the input to l2 is obtained via p3 not directly from l1 (without becoming the critical path). At power on, the output of l1 is initialised high and the output of l2 low.

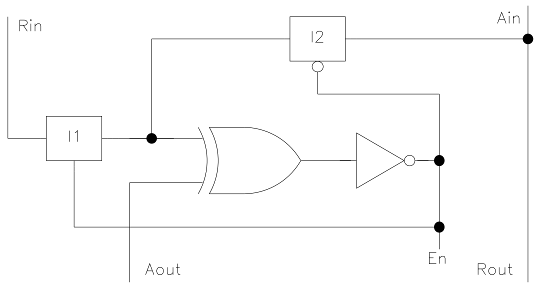

Fig. 5: Conceptual model

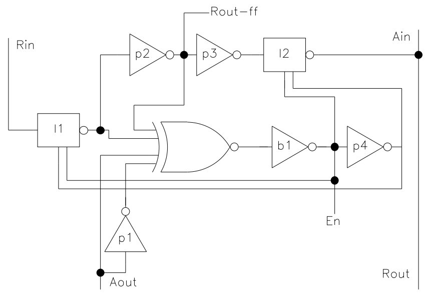

Fig. 6: Actual implementation

A modelling study of asynchronous latch controllers was carried out using the process algebra CCS [7] with its supporting tool, the Edinburgh Concurrency Workbench [8]. The results confirm that the proposed circuit of Fig. 5 operates as intended without timing assumptions other than those stated above.

IV Comparison

To evaluate the circuits, SPICE netlists (without layout extraction, ES2 5V 0.7μm) for a six stage FIFO were constructed. The left and right-hand sides supply and read data as fast as the pipeline will allow, such that the pipeline determines the speed. This corresponds to the method used in [3] rather than [2]. Capacitive loading is placed on En to simulate a 32-bit single-phase latch [2, Fig.9]. The two-phase circuits with the toggle element employ the improved toggle circuit given in [4]. The four-phase circuits compared are the semi-decoupled circuit [2, Fig.12], prior to the revision in [3], and the fully-decoupled circuit [3, Fig.13]. The simple four-phase controller in [2] is not included, as it does not allow adjacent stages to simultaneously hold data.

Table 1 lists the results. The cycle time was measured as the delay between requests from one stage to another. All values are for a single pipeline stage. Normalised values are shown in parentheses. The transistor count excludes devices used as resistive elements in the weak inverters (these are shown in parentheses). ‘FF’ indicates a fast-forward version.

Circuit	transistors	energy (pJ)	cycle-time (ns)	energy-delay
Fig. 5 FF	29 (4)	12.6 (1.00)	4.17 (1.00)	(1.00)
Fig. 3 FF	48 (2)	14.1 (1.12)	4.92 (1.18)	(1.32)
[2, Fig.12]	18 (4)	13.0 (1.03)	4.99 (1.20)	(1.23)
Fig. 5	29 (4)	12.8 (1.02)	5.63 (1.35)	(1.37)
Fig. 3	48 (2)	14.2 (1.13)	6.46 (1.55)	(1.75)
[3, Fig.13]	42 (8)	17.8 (1.41)	5.54 (1.33)	(1.88)

Table 1: Results of comparison

The four-phase circuits generate `Rout` excluding the data latch buffer delay time, so for a fair comparison they should be compared with the fast-forward two-phase circuits. The results indicate that the proposed two-phase circuits are faster, smaller and consume less power than the equivalent two-phase circuits and compare favourably with the four-phase circuits. The proposed two-phase circuit is inherently fully-decoupled (because it is two-phase) and so it would seem preferable to the fully-decoupled four-phase circuit when data processing logic is inserted between micropipeline stages.

V Conclusion

In this paper we have presented a latch controller for use in two-phase micropipelines in which the C-element and toggle element are replaced with two latches. The proposed circuit compares favourably with alternative two-phase and four-phase controllers.

References

[1] I. E. Sutherland, “Micropipelines,” in *Communications of the ACM*, vol. 32, pp. 720–738, June 1989.

[2] P. Day and J. V. Woods, “Investigation into micropipeline latch styles,” in *IEEE Transactions on VLSI Systems*, vol. 3, pp. 264–272, June 1995.

[3] S. Furber and P. Day, “Four-phase micropipeline latch control circuits,” in *IEEE Transactions on VLSI Systems*, vol. 4, pp. 247–253, June 1996.

[4] K. Y. Yun, P. A. Beerel, and J. Arceo, “High-performance two-phase micropipeline building blocks: double edge-triggered latches and burst-mode select and toggle circuits,” in *IEE Proc.-Circuits Devices Syst.*, vol. 143, pp. 282–288, October 1996.

[5] M. B. Josephs, “Event register specification and decomposition,” in *AMULET1 modelling workshop, Windermere*, July 1994.

[6] I. W. Jones, “Latch control circuits,” in *First International Symposium on Advanced Research in Asynchronous Circuits and Systems, Utah*, November 1994.

[7] R. Milner, *Communication and Concurrency*. Prentice Hall, 2nd ed., 1989.

[8] R. Cleaveland, J. Parrow, and B. Steffen, “The concurrency workbench,” vol. 407 of *LNCS*, (Berlin), pp. 24–37, Springer, June 1990.

Acknowledgements
We would like to thank Graham Clark for his help with the CCS modelling. This work was supported by EPSRC award 95305666.