Benefits of Asynchronous Control for Analog Electronics: Multiphase Buck Case Study

Danil Sokolov†, Vladimir Dubikhin†, Victor Khomenko†, David Lloyd‡, Andrey Mokhov†, Alex Yakovlev†
†Newcastle University, UK; ‡Dialog Semiconductor, UK

***Abstract*—*Analog and mixed signal* (AMS) electronics becomes increasingly complex and needs to be digitally enhanced by its own control circuitry. The RTL synthesis flow routinely used for digital logic is however optimized for synchronous data processing and produces inefficient control for AMS. In this paper we demonstrate the evident benefits of asynchronous circuits in the context of AMS systems, and propose an *asynchronous design for analog electronics* (A4A) flow for their specification, synthesis, and formal verification. A library of specialized *analog-to-asynchronous* (A2A) components is developed for interfacing analog signals to asynchronous control. A4A flow is automated in the WORKCRAFT framework and evaluated using a multiphase buck converter case study. The simulation results show improved response time, voltage ripple, and peak current of the buck when controlled asynchronously. These benefits lead to the higher efficiency of power conversion, and can be traded off for the cost of analog components. A4A flow, A2A interfaces, and WORKCRAFT tools are used for development of power converters at Dialog Semiconductor.**

I. MOTIVATION

The complexity of modern systems on chip is rapidly growing, much as the role of *analog and mixed signal* (AMS) electronics which provides an important infrastructure for distributing and regulating energy flows, monitoring the system's operating conditions, and interfacing with the continuous non-digital environment, see Figure 1. The complexity of AMS itself is also increasing to accommodate the switching dynamics, process variability, and reliability requirements of heterogeneous multi-core systems and emerging IoT devices [1]. As a result, the analog layer needs to be "digitally enhanced" [2] with its own "little digital" control that efficiently operates at different *time bands*: local (within digital domain), fast (e.g. fetch the best available sensor reading), and slow (e.g. activate an analog component and wait for the change at a sensor). Power converters [3] are of particular importance as energy is becoming the most valuable resource in modern electronics.

The responsiveness and robustness of power converters is determined by the implementation of their control circuitry: millions of control decisions need to be made every second and a single incorrect decision may cause a malfunction of the whole system or even permanently damage the circuit. For example, a 3MHz switching regulator is clocked around 473,364,000,000,000 times in 5 years of its operation [4].

A practical design problem associated with the development of digital logic within power converters [5] is partially related to the state-of-the-art synthesis methods: the conventional RTL flow is neither aimed at nor suited for building "little digital" controllers. It is primarily targeted at building high throughput,

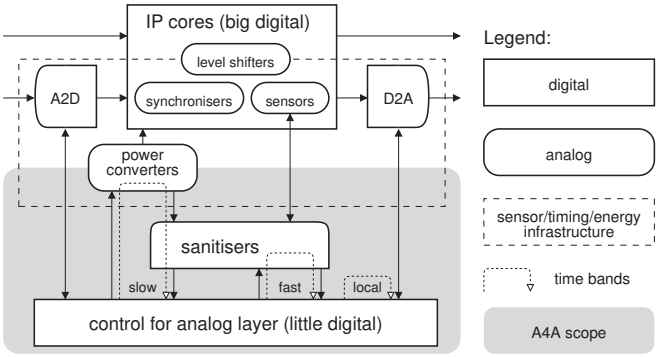


Figure 1: Asynchronous design for analog electronics.

often pipelined, data processing logic. The global clocking paradigm imposed by RTL leads to either low responsiveness or power consumption overheads for AMS control. There is a fundamental contradiction in terms of timing requirements for such a control. On one hand, the clock frequency must be sufficiently high to promptly react to changes at the analog inputs (e.g. sensor readings). On the other hand, this leads to massive waste of energy due to useless switching of the global clock and "sanitizing" the readings (typically, by sampling and synchronizing) when the environment changes slowly. Moreover, the probability of failure (due to metastability conditions lasting longer than a clock period) in synchronizers becomes a significant factor in systems' reliability when the number of asynchronous signals and the clock frequency increase. Hence, there is an ever increasing scope for the use of *asynchronous design for analog electronics* (A4A). AMS control can significantly benefit from the use of asynchronous logic [6], [7] that does not rely on the global clocking and operates at the pace determined by the operating conditions. Such circuits are adaptable to the rate of changes in the controlled system and can react to the asynchronous signals from the sensors without the need for synchronizers.

There are many methodologies and design styles for asynchronous circuits [8]. However, to the best of our knowledge, none of them tackles the problems of design automation for "little digital" control. There is no consistent framework for efficient and reliable interfacing between the digital and analog worlds. Existing solutions are often *ad hoc* and based on unrealistic assumptions about the behaviour of the non-persistent outputs of analogue comparators and sensors, which may lead to hazards propagating into the digital core of the system. In this paper we address these challenges.

978-3-9815370-8-6/17/$31.00

The main contributions of this paper are as follows:

- A **novel A4A design flow** for the systematic development of "little digital" asynchronous controllers.
- **Methodology for deployment of A2A components** to efficiently and reliably interface non-persistent inputs.
- Automation of A4A flow in **WORKCRAFT toolkit** [9].
- Demonstration of **clear measurable benefits** of asynchronous controllers over the traditional synchronous ones using a multiphase buck converter as a case study.

Dialog Semiconductor uses the WORKCRAFT toolkit and A2A components for designs that have been used on production chips, which demonstrates the maturity of the A4A flow.

II. BUCK CONVERTER

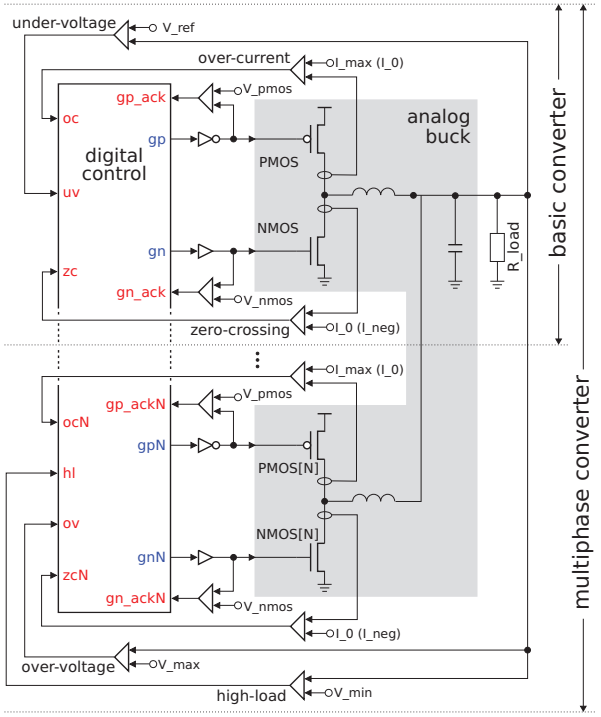
Power regulation is one of the areas of AMS design that needs to be enhanced with a "little digital" control. We use a multiphase buck converter case study to demonstrate the advantages of asynchrony and A4A design methodology.

A buck converter comprises an analog buck and a digital control [3], as shown in Figure 2a. In *basic* case the control operates a pair of power regulating PMOS and NMOS transistors of the buck (using gp and gn outputs) as a reaction to *under-voltage* (UV), *over-current* (OC) and *zero-crossing* (ZC) conditions (uv, oc and zc inputs respectively). These conditions are detected by sensors that compare the measured current and voltage with some reference values (V_ref, I_max, I_0). Note that in order to avoid a short-circuit the PMOS and NMOS transistors of the buck must never be ON at the same time. Therefore, the controller is explicitly notified (by the gp_ack and gn_ack signals) when the power transistor threshold levels (V_pmos and V_nmos) are crossed.

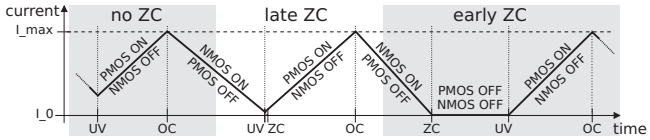
The operation of a power regulator is usually specified in an intuitive, but rather informal way, e.g. by enumerating the possible sequences of detected conditions and describing the intended reaction to these events, as shown in Figure 2b. The diagram shows that UV should be handled by switching the NMOS transistor OFF and PMOS transistor ON, while OC should revert their state – PMOS OFF and NMOS ON (no ZC scenario). Detection of the ZC after UV does not change this behaviour (late ZC scenario). However, if ZC is detected before UV then both the PMOS and NMOS transistors remain OFF until the UV condition (early ZC scenario).

A *multiphase* buck converter combines several pairs of PMOS and NMOS transistors (called *phases*) to power the same load, see Figure 2a. The main advantages of this distributed design compared to the basic buck are faster reaction to the power demand, heat dissipation from a larger area, and the possibility to replace a large coil with several smaller ones, thus reducing the dimensions of consumer gadgets [3].

The control circuit of a multiphase buck with N phases monitors the OC and ZC conditions of individual phases (inputs oc1,...,ocN and zc1,...,zcN) and the voltage level at the load (hl, uv and ov inputs). When UV is detected (the voltage drops below V_ref value) the controller performs a charging cycle (switching the PMOS and NMOS transistors the same way as in the basic buck) at the currently active phase.



(a) Schematic.



(b) Informal specification of a basic buck.

Figure 2: Buck converter.

The active phase is traditionally selected in a round-robin pattern by a generator of non-overlapping pulses. If by the time the next phase is activated the UV condition still persists, a charging cycle is exercised by that phase too, thus helping the previous phase(s). This process is repeated until the power demand is met and the UV condition is reset, and is resumed upon detection of the next UV. A special mode of operation is used to handle the high-load (HL) condition that indicates a sudden increase in power demand (the voltage drops below V_min value). In this mode the controller activates all the phases simultaneously and, as HL implies UV (V_min < V_ref), a charging cycle starts in all the phases.

As buck rumps to its target voltage it can overshoot (the voltage goes above V_max value) and enter the over-voltage (OV) mode to sink excessive energy. This is usually achieved by changing the reference values for OC and ZC conditions (I_0 and I_neg values respectively), so that PMOS is switched OFF as soon as positive current is detected and NMOS stays ON until the negative current limit is reached.

For efficiency reasons, once ON, the PMOS and NMOS transistors should not switch OFF for at least the predefined PMIN and NMIN time intervals, respectively. Furthermore, the minimum ON time of PMOS transistor is extended by PEXT at the first charging cycle upon detection of UV condition.

III. A4A DESIGN FLOW

Development of A4A control is a complex process with multi-dimensional optimisation possibilities and verification challenges that requires design automation. Our proposed A4A design flow is shown in Fig. 3. It starts with an informal specification of intended system behaviour (in form of phase diagrams, waveforms, and verbal requests). This needs to be formalised in a consistent and unambiguous form of *signal transition graphs* (STGs) – a special type of Petri nets whose transitions are associated with rising and falling edges of signals [10]. This step is the most difficult to automate and relies on designer experience and established design guidelines for decomposition of the design into simple sub-modules.

The obtained specifications of the sub-modules are synthesised into speed-independent gate-level components [10], which are integrated into a complete "little digital" controller. Standard EDA tools can be reused for place-and-route and offline testing of asynchronous "little digital" controllers [11].

Verification is applied at every stage of A4A flow: for sanity checks of the formal specification (e.g. deadlock-freeness and consistency of signals), for functional correctness of the gate-level implementation (e.g. conformance to the specification and absence of hazards), and for timing verification of the complete system (e.g. validation of the timing assumptions).

For interacting with the analog components the asynchronous controller relies on a library of A2A interface elements for sanitising non-persistent inputs (e.g. those coming from voltage comparators), e.g. see Fig. 5c. The core A2A components and their functionality are as follows:

- WAIT element waits for the non-persistent input to become high and then latches it until explicitly reset through the asynchronous output handshake. The non-persistent behaviour and associated metastability is fully contained within the element, guaranteeing a clean speed-independent asynchronous output. This is a basic A2A interface which is used when implementing other, more sophisticated interfaces. The symmetric element that waits for the input to become low is called WAIT0. See [16] for implementation details.
- WAIT2 is a combination of WAIT and WAIT0 elements: it uses a 2-phase output handshake, waiting for high and low input values, one after the other.
- RWAIT and RWAIT0 are modifications of the WAIT and WAIT0 elements, respectively, with a possibility to persistently cancel the waiting request. This is useful when the input is no longer expected to change or the change is no longer relevant for the asynchronous controller, and hence the output handshake needs to be released.
- WAIT01 and WAIT10 elements wait for a rising or falling edge of the input signal, respectively. Note, this is subtly different from waiting for high or low input value, e.g. a signal can be initially low, and to generate a falling edge event it must first go high.
- WAITX element is used to arbitrate between two non-persistent inputs. It isolates the asynchronous controller both from the metastability associated with non-persistent analog inputs, as well as from the metastability associated with making the decision of which signal goes high first, providing a clean dual-rail output interface to the asynchronous controller.
- WAITX2 behaves as WAITX in the rising phase and as WAIT0 in the falling phase, i.e. it does not release the output asynchronous handshake until the winning input signal goes low. It uses a 2-phase output handshake similarly to WAIT2.

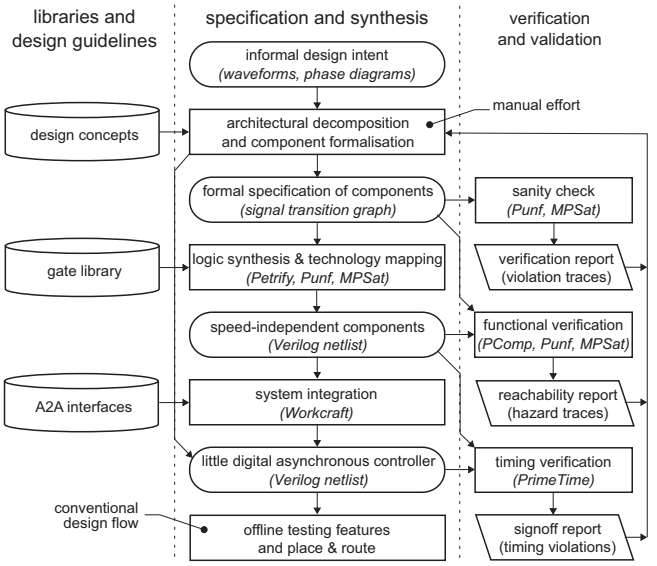


Figure 3: A4A design flow.

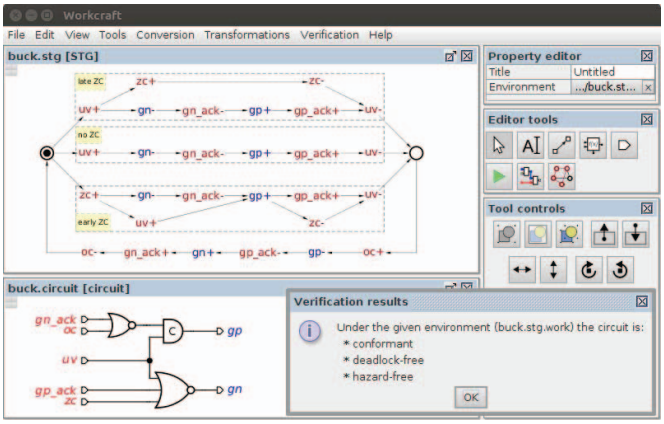
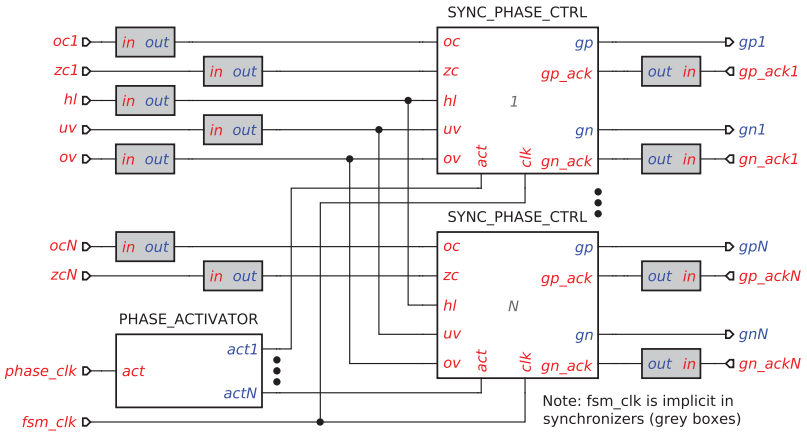
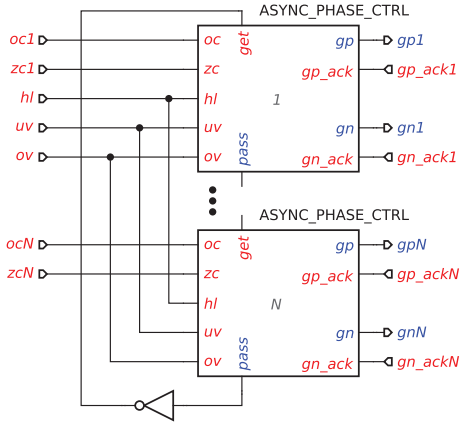


Figure 4: Development of basic buck in WORKCRAFT.

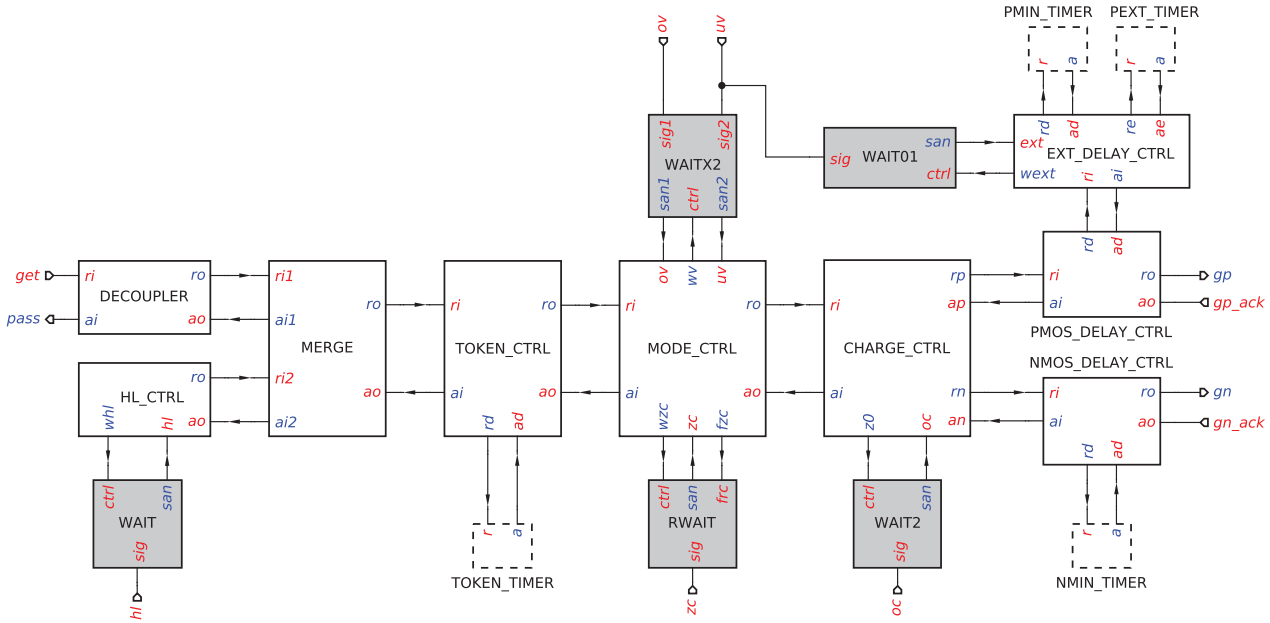
The A4A design flow is automated in WORKCRAFT framework [9]. It reuses established backend tools, such as PETRIFY [12] and MPSAT [13], for logic synthesis and formal verification tasks. The screenshot in Fig. 4 illustrates development of the basic buck controller: its STG specification (top) is automatically synthesised as a speed-independent circuit (bottom) and formally verified (the verification results window). In case of a violation a trace leading to the problematic state would be reported for interactive simulation in WORKCRAFT to conveniently localise and debug the issue [14].



(a) Synchronous round robin architecture.



(b) Asynchronous token ring architecture.



(c) Decomposition of an asynchronous phase controller.

Figure 5: Multiphase buck controller.

IV. Design of Multiphase Buck Controller

This section presents synchronous and asynchronous controllers for the multiphase buck and overviews their most interesting design aspects. Some details have been omitted to shorten the presentation or due to commercial sensitivity.

Synchronous control

A top-level architecture of synchronous N-phase buck controller is shown in Figure 5a. It consists of N phase control modules (one per buck phase) and a phase activator that selects the phase controllers in a round-robin pattern. Sanitising the asynchronous non-persistent outputs of the buck sensors is done by the synchronizers [15] (shaded components). The other modules operate on clean digital signals and can be specified in conventional RTL style as clocked FSMs and synthesised by the standard EDA flow.

Two global clocks are used in this design: phase_clk – a relatively slow clock (few MHz) for generating non-overlapping pulses to activate the phases, and fsm_clk – a fast clock (hundreds of MHz) for polling the sensors and clocking the FSM. The latter clock is implicit in all synchronisers (not shown for clarity). Note that synchronization (e.g. using 2-flop synchronizers) imposes a latency of up to 2.5 clock periods[1] in the reaction time of the synchronous buck controller.

Asynchronous control

Asynchronous control for the multiphase buck is built as a token ring with several identical phase controller stages [16],

[1]Two for synchronisation and 0.5 for FSM operation – the reduction of 0.5 clock period can be achieved by doing the synchronization on the negative clock edge and the FSM computation on the positive one. In the worst case, if a synchronizer hits metastability, the latency may increase by another clock period or even result in a synchronization failure.

as shown in Figure 5b. Absence of global clocking enables its communication with the fundamentally asynchronous environment without the need for synchronizers. Each stage delays the token for at least a predefined duration of time (corresponds to the period of the slow phase activation clock in synchronous design) before propagating it to the next stage. As the token enters a stage, this stage becomes active and will eventually perform a charging cycle at the corresponding buck phase.

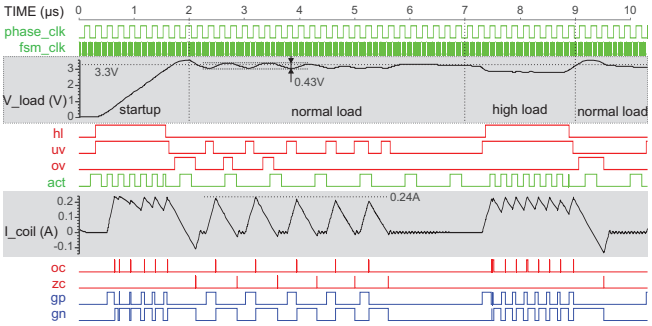
A single phase of the controller is shown in Figure 5c. It is partitioned into sub-modules to simplify the process of specification and reduce the synthesis and verification effort. The sub-modules communicate by means of handshakes with the following naming convention: requests start with 'r' and acknowledgements with 'a'. The second letter refines their semantics – 'i'/'o' for input and output channels, 'd' for timer interfaces, and 'p'/'n' for PMOS and NMOS power transistors.

In this design the sensor readings are sanitised using A2A interface components (shaded), thus waiting for the specific change of a non-persistent input rather than continuously polling its state as in the synchronous design. E.g. a WAIT is used to wait for the HL condition, a WAITX2 to identify the UV and OV modes (though these modes are mutually exclusive in theory, switching between them can happen fast, and so one has to arbitrate between them), an RWAIT to wait for ZC condition (it can be reset due to a timeout), and a WAIT2 to monitor the state of the OC condition.

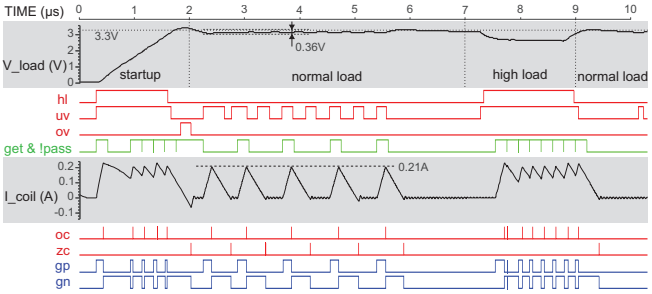
The phase controller performs two distinct functions, handling its activation and charging the buck. The stage may become active either when it receives a token from the previous stage (get/pass interface to the DECOUPLER component) or when the HL condition is detected by HL_CTRL. The OR-causality between these scenarios is handled by the MERGE component which is implemented using the *opportunistic merge* element [17]. TOKEN_CTRL starts TOKEN_TIMER to delay passing the token to the next phase and simultaneously activates MODE_CTRL that monitors the UV and OV conditions to determine the mode of buck operation. Note that HL implies UV, as they both are the results of comparisons of the same voltage with different thresholds. We exploit this in our design: when a stage is activated by the HL condition, the charging is still initiated by UV as in the regular case. MODE_CTRL also decouples token propagation from charging by giving an early acknowledgement to TOKEN_CTRL immediately after either the UV or OV condition is detected.

CHARGE_CTRL conducts a cycle of charging of a buck phase following a pattern similar to that of the basic buck. PMOS_DELAY_CTRL and NMOS_DELAY_CTRL enforce the requirement for the minimum ON time for PMOS and NMOS transistors by delaying the corresponding acknowledgements. They employ PMIN_TIMER and NMIN_TIMER to specify the delays. To keep PMOS transistor ON longer on the first cycle of charging in the UV mode (detected by the WAIT01), EXT_DELAY_CTRL uses PEXT_TIMER.

STG specifications of all controller modules were developed, synthesised and verified using WORKCRAFT framework [9], [14]. We verified that all STGs are consistent, deadlock-free, and output-persistent. We also verified specific buck converter properties, such as the absence of a short circuit in PMOS/NMOS transistors and the possibility of sharing some of the timers. All the gate-level implementations were also verified to be deadlock-free, hazard-free and conformant to their STG specifications.



(a) Synchronous control at 333MHz.



(b) Event-driven asynchronous control.

Figure 6: Simulation waveforms.

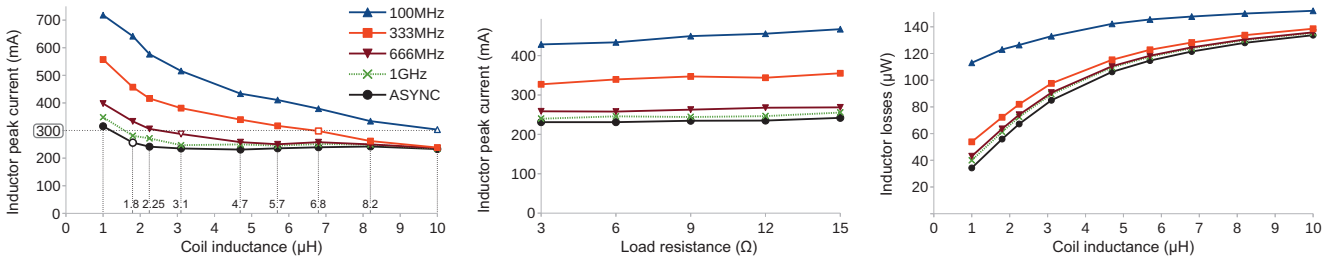
Controller	HL (ns)	UV (ns)	OV (ns)	OC (ns)	ZC (ns)
100MHz	25.00	25.00	25.00	25.00	25.00
333MHz	7.50	7.50	7.50	7.50	7.50
666MHz	3.75	3.75	3.75	3.75	3.75
1GHz	2.50	2.50	2.50	2.50	2.50
ASYNC	1.87	1.02	1.18	0.75	0.31
Improvement over 333MHz	4x	7x	6x	10x	24x

Table I: Comparison of the reaction time.

V. EXPERIMENTAL RESULTS AND ANALYSIS

We implemented a 4-phase buck with synchronous and asynchronous controllers. The analog components were modelled in VERILOG-A and the digital controllers were implemented in TSMC 90nm technology. Synchronous controller was synthesised using SYNOPSYS DESIGNCOMPILER for 100MHz, 333MHz, 666MHz, and 1GHz. Response time of synchronous control is 2.5 clock periods, as explained in Section IV. The latency of asynchronous design was measured in SYNOPSYS PRIMETIME.

The operation of the buck was validated and its efficiency was estimated by simulation with CADENCE INCISIVE using an AMS testbench. Coils were modelled in the range from 1µH to 10µH using the parameters of COILCRAFT RF inductors [18]. Figure 6 shows the simulation waveforms for one of the buck phases. One can notice that the asynchronous buck



(a) Peak current for 1-10μH coils at 6Ω load. (b) Peak current for 3-15Ω loads at 4.7μH coil. (c) Losses for 1-10μH coils at 6Ω load.

Figure 7: Comparison of peak current and inductor losses.

enjoys smaller voltage overshoot after resolving the first HL condition at buck *startup* (1-2μs). This results in shorter OV resolution time and absence of recurring OV conditions, which can be observed in the synchronous buck waveform (2-4μs). Note that asynchronous buck does not even overshoot at the exit from *high load* (7-8μs). At *normal load* (2-7μs) asynchronous buck demonstrates smaller voltage ripple and lower inductor peak current than the synchronous buck: 0.36V vs 0.43V and 0.21A vs 0.24A, respectively. These advantages are due to faster reaction of the asynchronous controller to the input stimuli (HL, UV, OV, OC, and ZC conditions), as summarised in Table I – synchronous control has constant latency of 2.5 clock cycles while asynchronous control exhibits significantly faster path-dependent reaction. To achieve response times similar to the asynchronous controller, the synchronous circuit would need to be clocked at ~3GHz, which requires expensive deep-submicron fabrication process and makes the design challenging.

The quick response of the asynchronous control enables it to operate with a significantly smaller peak current when using the same coils, see Figure 7a. This advantage can be efficiently traded off for the size of coils, which are bulky and affect the dimensions of consumer gadgets. For example, for a 6Ω load, the asynchronous control maintains the peak current below 300mA using 1.8μH inductors, while the synchronous control requires 10μH coils at 100MHz, 6.8μH at 333MHz, or 3.1μH at 666MHz (denoted by hollow markers in Figure 7a). This trend persists for a wide range of load resistance that covers the typical computational load of mobile microprocessors, see Figure 7b for the peak current data at 3-15Ω loads and 4.7μH coils. The smaller coil inductance also translates into fewer losses, as shown in Figure 7c, and helps to achieve higher power efficiency. Therefore it is advantageous to choose the smallest possible coil to reduce the inductor losses and physical dimensions of the system, while maintaining its operating characteristics.

VI. Conclusion

This work demonstrates clear advantages of asynchronous design methodology for "little digital" control. The simulation results show improved reaction time, voltage ripple, peak current, and inductor losses of the buck when controlled asynchronously. These benefits lead to the higher efficiency of power conversion, and can be traded off for the cost of analog components. A4A design flow, A2A interfaces, and WORKCRAFT tools are used at Dialog Semiconductor.

Acknowledgements

This research was supported by EPSRC grants EP/L025507/1 (A4A) and EP/K001698/1 (UNCOVER). A PhD scholarship for Vladimir Dubikhin was sponsored by Dialog Semiconductor.

References

[1] A. Talbot: *"Holistic mixed signal design in ultra deep sub-micron technologies"*, NMI R&D Workshop: AMS Design, 2016.
[2] B. Murmann, C. Vogel, H. Koeppl: *"Digitally enhanced analog circuits: system aspects"*, Proc. Int. Symp. on Circuits and Systems (ISCAS), pp. 560–563, 2008.
[3] A. Pressman, K. Billings, T. Morey: *"Switching power supply design"*, 3rd edition, McGraw-Hill, 2009.
[4] J. Audy: *"Navigating the path to a successful IC switching regulator design"*, Tutorial at Int. Solid-State Circuits Conference (ISSCC), 2008.
[5] T. Towers: *"Practical design problems in transistor DC/DC converters and DC/AC inverters"*, Proc. IEE, vol. 106(18), pp. 1373–1383, 1959.
[6] S. Unger: *"Asynchronous sequential switching circuit"*, Wiley-Interscience, 1969.
[7] D. Muller, W. Bartky: *"A theory of asynchronous circuits"*, Proc. Int. Symp. of the Theory of Switching, pp. 204–243, 1959.
[8] J. Sparsø, S. Furber: *"Principles of asynchronous circuit design"*, Kluwer Academic Publishers, 2001.
[9] WORKCRAFT homepage: http://workcraft.org/.
[10] T.-A. Chu: *"Synthesis of self-timed VLSI circuits from graph-theoretic specifications"*, PhD thesis, Massachusetts Institute of Technology, 1987.
[11] D. Lloyd, R. Illman: *"Scan insertion and ATPG for C-gate based asynchronous designs"*, Synopsys User Group (SNUG), 2014.
[12] J. Cortadella, M. Kishinevsky, A. Kondratyev, L. Lavagno, A. Yakovlev: *"Petrify: a tool for manipulating concurrent specifications and synthesis of asynchronous controllers"*, IEICE Transactions on Information and Systems, vol. E80-D(3), pp. 315–325, 1997.
[13] V. Khomenko, M. Koutny, A. Yakovlev: *"Logic synthesis for asynchronous circuits based on STG unfoldings and incremental SAT"*, Fundamenta Informaticae, vol. 70(1-2), pp. 49–73, 2006.
[14] I. Poliakov, A. Mokhov, A. Rafiev, D. Sokolov, A. Yakovlev: *"Automated verification of asynchronous circuits using circuit Petri nets"*, Proc. Asynchronous Circuits and Systems (ASYNC), pp. 161–170, 2008.
[15] D. Kinniment: *"Synchronization and Arbitration in Digital Systems"*, Wiley Publishing, 2008.
[16] D. Sokolov, V. Khomenko, A. Mokhov, A. Yakovlev, D. Lloyd: *"Design and verification of speed-independent multiphase buck controller"*, Proc. Asynchronous Circuits and Systems (ASYNC), pp. 29–36, 2015.
[17] A. Mokhov, V. Khomenko, D. Sokolov, A. Yakovlev: *"Opportunistic merge element"*, Proc. Asynchronous Circuits and Systems (ASYNC), pp. 116–123, 2015.
[18] *"Modeling Coilcraft RF Inductors"*, http://www.ing.unp.edu.ar/electronica/asignaturas/ee016/anexo/l-coilcraft-pspice.pdf