FlexiCores: Low Footprint, High Yield, Field Reprogrammable Flexible Microprocessors

Nathaniel Bleier
University of Illinois
Urbana, Illinois, USA
nbleier3@illinois.edu

Calvin Lee
University of Illinois
Urbana, Illinois, USA
clee3@illinois.edu

Francisco Rodriguez
PragmatIC Semiconductor
Milton, Cambridge, UK

Antony Sou
PragmatIC Semiconductor
Milton, Cambridge, UK

Scott White
PragmatIC Semiconductor
Milton, Cambridge, UK

Rakesh Kumar
University of Illinois
Urbana, Illinois, USA
rakeshk@illinois.edu

ABSTRACT

Flexible electronics is a promising approach to target applications whose computational needs are not met by traditional silicon-based electronics due to their conformality, thinness, or cost requirements. A microprocessor is a critical component for many such applications; however, it is unclear whether it is feasible to build flexible processors at scale (i.e., at high yield), since very few flexible microprocessors have been reported and no yield data or data from multiple chips has been reported. Also, prior manufactured flexible systems were not field-reprogrammable and were evaluated either on a simple set of test vectors or a single program. A working flexible microprocessor chip supporting complex or multiple applications has not been demonstrated. Finally, no prior work performs a design space exploration of flexible microprocessors to optimize area, code size, and energy of such microprocessors.

In this work, we fabricate and test hundreds of FlexiCores - flexible 0.8 µm IGZO TFT-based field-reprogrammable 4 and 8-bit microprocessor chips optimized for low footprint and yield. We show that these gate count-optimized processors can have high yield (4-bit FlexiCores have 81% yield - sufficient to enable sub-cent cost if produced at volume). We evaluate these chips over a suite of representative kernels - the kernels take 4.28 ms to 12.9 ms and 21.0 µJ to 61.4 µJ for execution (at 360 nJ per instruction). We also present the first characterization of process variation for a flexible processor - we observe significant process variation (relative standard deviation of 15.3% and 21.5% in terms of current draw of 4-bit and 8-bit FlexiCore chips respectively). Finally, we perform a design space exploration and identify design points much better than FlexiCores - the new cores consume only 45-56% the energy of the base design, and have code size less than 30% of the base design, with an area overhead of 9-37%.

ISCA '22, June 18–22, 2022, New York, NY, USA

ACM ISBN 978-1-4503-8610-4/22/06…$15.00
https://doi.org/10.1145/3470496.3527410

ACM Reference Format:
Nathaniel Bleier, Calvin Lee, Francisco Rodriguez, Antony Sou, Scott White, and Rakesh Kumar. 2022. FlexiCores: Low Footprint, High Yield, Field Reprogrammable Flexible Microprocessors. In *The 49th Annual International Symposium on Computer Architecture (ISCA '22), June 18–22, 2022, New York, NY, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3470496.3527410

1 INTRODUCTION

As incredible as it sounds, the reach of computing, while impressive, is still limited. A vast set of applications with ultra-low-cost, thinness, and conformality requirements – for example wearable patches [21], garments [78], packaging [29] for fast moving consumer goods (FMCG), low-end healthcare products such as smart bandages [14], and disposable sensors for food [9], pharmaceuticals [2], agriculture and forestry [86], and environment [60] – have not seen much penetration of computing.

The primary reason is that today's computing is based on silicon, but silicon-based electronics has inherent limitations in terms of thinness [16], conformality [38], and cost [37]. For example, the inherently high processing cost of metal–oxide–semiconductor field-effect transistors (MOSFETs) fabricated on crystalline silicon wafers have prevented sub-cent costs of silicon-based processing devices in spite of economies of scale. This prevents targeting applications such as item-level tagging, which require sub-cent costs [76]. Similarly, many applications have thinness and conformality requirements [16] that cannot be naturally met by silicon.

Flexible electronics is a promising approach to target applications whose computational needs cannot be met by traditional electronics due to their conformality, thinness, or cost requirements [4, 5]. Flexible electronic devices are built on flexible substrates (e.g., paper [50], plastic [101], or metal foil [52]) using active thin-film semiconductor materials (e.g., organic [50] or metal oxide [49], or amorphous-silicon [77]). Although traditional MOSFETs outperform thin-film transistors (TFTs) in power, area, and performance [51, 83], TFTs have much lower fabrication cost [91, 93, 100]; also, the corresponding substrates are naturally thin, flexible, and low cost. This makes flexible electronics an enticing technology for these applications [3, 12, 27, 57, 98].

In this work, we focus on flexible microprocessors. The cost benefits and programability of microprocessors make them a critical electronic component even for the applications we target [4]. There

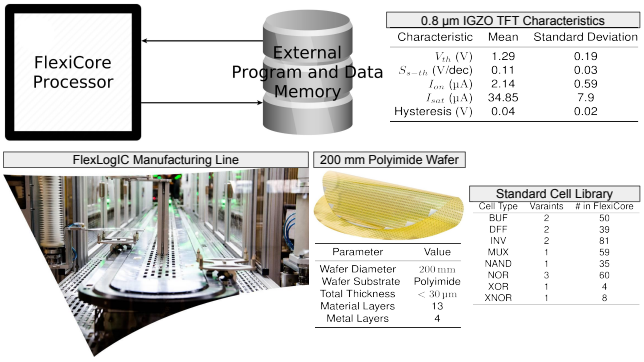


Figure 1: A FlexiCore flexible processor reads and executes an arbitrary program resident in an external program memory and can interact with external input and output devices. FlexiCore designs, described in Verilog HDL, were synthesized to a 0.8 µm IGZO cell library and fabricated using PragmatIC's commercial 'fab-in-a-box' manufacturing line, FlexLogIC. FlexiCore is the first field-reprogrammable flexible microprocessor.

have been several attempts at integrating thinned silicon microprocessor dies on to flexible substrates [59, 88]. While this hybrid integration can sometimes meet conformality requirements, cost continues to be a limitation [92] due to the high cost of silicon manufacturing (integration costs increase as well). What the targeted applications really need are low footprint and high yield (and, therefore, low cost) natively flexible microprocessors.

Unfortunately, manufacturing natively flexible microprocessors has been a challenge and very few such processors have been reported [4, 35, 69] (Section 2). This is not surprising since processors typically require integration of a relatively large number of devices, which is hard in the emerging TFT technology. In the small number of cases where a flexible microprocessor has been successfully manufactured and tested, no yield data has been reported. Data from multiple chips has not been reported either for a given flexible microprocessor. As a result, it is still unclear whether it is feasible to build natively flexible processors at scale (i.e., at high yield). In fact, to the best of our knowledge, *no prior non-silicon processor has been shown to yield at scale.*

Prior manufactured flexible systems were not field reprogrammable and were tested either on a simple set of test vectors or a single program. A working flexible microprocessor chip supporting complex or multiple applications has not been demonstrated (Section 5).

Finally, no prior work looks carefully at the design space of flexible microprocessors to understand how different architectural and microarchitectural optimizations impact area, code size, and energy of such microprocessors. As such, it is unclear if flexible microprocessors should support accumulator-based ISA or load-store ISA, single cycle, multi-cycle or pipelined implementation, certain set of operations, etc.

In this work, we fabricate and test hundreds of flexible processor chips to establish the feasibility of building low footprint, high yielding, field reprogrammable flexible processors. We observe that while recent work [4] demonstrated a 32-bit Indium-Gallium-Zinc-Oxide (IGZO)-based natively flexible Cortex-M microprocessor, many use cases for flexible electronics do not require the performance and expressiveness of a Cortex-M core (Section 5). Therefore, instead of building a relatively complex, 32-bit Cortex-M-like circuitry with high gate counts, we focus on low levels of computation with simple 4-bit and 8-bit architectures optimized for footprint and yield, so that low-cost points can be achieved in volume production.

We design and fabricate three flexible processors - FlexiCores. We optimized the architecture and micro-architecture of FlexiCores for these attributes through a careful selection of ISA, and are the first to report high yield levels suitable for commercial volume production. In fact, this may be the first demonstration of any non-silicon processor at scale.

Furthermore, since our processors are field reprogrammable, we demonstrate performance and energy characteristics of FlexiCore processors over a suite of kernels, including relatively complex kernels representing sensor processing, control, and machine learning applications. Our kernels take 4.28 ms to 12.9 ms and 21.0 µJ to 61.4 µJ for execution (at 360 nJ per instruction). This is the first demonstration of a working flexible microprocessor chip supporting multiple applications.

We also present the first characterization of process variation for a flexible processor. We find that flexible processors demonstrate significant process variation (relative standard deviation of 15.3% and 21.5% in terms of current draw of 4-bit and 8-bit FlexiCore chips respectively).

We leverage our experience with FlexiCores to set up a design space exploration for flexible microprocessors. We explore how different architectural and microarchitectural decisions impact trade-offs between area, code size, and energy. Our exploration yields design points that are considerably more efficient than FlexiCores. The resulting cores consume only 45-56% the energy of the base design, and have code size less than 30% of the base design, with an area overhead of 9-37%.

This paper makes the following contributions:

- We design, fabricate, and test FlexiCore chips - natively flexible 4-bit and 8-bit microprocessors optimized for low footprint and yield.
- We show that it is feasible to manufacture natively flexible microprocessors at high yield, provided their architecture and microarchitecture have been optimized for low gate count.
- We show performance and energy data for FlexiCores for a suite of representative kernels. We also present process variation data for FlexiCores chips. These are first such studies for flexible microprocessors.
- We present a design space exploration for flexible processors and identify design points much better than FlexiCores.

2 RELATED WORK

Early work on flexible microprocessors was based on low-temperature poly-silicon TFTs [13, 42, 53, 87]. However, poly-silicon TFT technologies have high manufacturing costs and lateral device architecture scaling is limited to the > 1 µm range [67].

Table 1: Example applications and their performance/precision requirements

Application	Sample Rate (Hz)	Prec. (bits)	Duty Cycle Period	Application	Sample Rate (Hz)	Prec. (bits)	Duty Cycle Period
Blood Pressure Sensor [70]	< 100	< 8	Hours [19]	Body Temperature Sensor [70]	< 1	< 8	Minutes [44]
Odor Sensor [70]	16-25	< 8	Minutes [73]	Smart Bandage [65]	< 0.01	< 8	Continuous to Hours [23]
Heart Beat Sensor [70]	< 4	1	Seconds [90]	Tremor Sensor [33]	< 25	16	Seconds [20]
Pressure Sensor [31]	1-5.5	12	Continuous to Hours [81]	Oral-Nasal Airflow [70]	< 25	< 8	Seconds
Light Level Sensor [70]	< 1	< 8	Continuous to Hours [22]	Perspiration Sensor [47]	< 25	< 8	Minutes [99]
Trace Metal Sensor [47]	25	16	Minutes	Pedometer [72]	< 25	1	Seconds [72]
Food Temp. Sensor [70]	< 1	< 8	5 minutes [82]	Timer [40]	1	1	Single Use
Alcohol Sensor [48]	1	< 8	Single Use [64]	POS Computation [63]	< 100	< 8	Single Use [63]
Humidity Sensor [34]	10	16	Continuous to Hours [80]	Smart Labels [7]	1	< 8	Seconds
Pseudo-RNG	n/a	< 8	Seconds	Error Detection Coding	< 100	< 8	Continuous to Hours

Recent work on flexible microprocessors includes a 16-bit RISC-V microprocessor built using flexible carbon-nanotube (CNT) TFTs [35]. a one-bit microprocessor [94] built using flexible molybdenum disulfide transistors. However, although the devices were flexible, the microprocessors themselves were fabricated on conventional silicon wafers, and thus the resulting ICs were not flexible.

Some recent work on flexible processors is based on metal-oxide TFTs. Such TFTs are low-cost and scale well and are, therefore, a good fit for building complex components such as microprocessors. Myny *et al.* [69] report an 8-bit ALU built using metal-oxide TFTs and integrated with a polyimide-based print-programmed ROM. Ozer *et al.* [74] report a flexible machine learning classifier. Biggs *et al* [4] report a 32-bit natively flexible ARM-v6m microprocessor.

Our work differs on multiple counts. First, no previous work has reported yield rate or data from multiple chips. We are the first to report yield information. We are also the first to report process variation data for flexible processors. Second, most previous efforts in flexible microprocessors have featured fixed or factory-programmable ROMs [4, 69], while FlexiCores can execute (and modify) programs stored in off-chip memories. This allows us to present the first performance and energy evaluation of a flexible microprocessor over a suite of representative kernels. Third, most prior work does not explicitly look to identify characteristics of a good ISA or microarchitecture for flexible processors. We explicitly optimize the ISA and the microarchitecture for low footprint and high yield. Furthermore, we perform a design space exploration to explicitly look for design points better than FlexiCores. This is the first such design space exploration for flexible processors.

There are other works on tiny, low gate count processors with high variation and high static power [17, 43, 71]. Our work differs in focus (flexible processors) and technology (0.8 µm IGZO). Also, our applications have much more relaxed performance constraints (Table 1). Our area constraints are also far more stringent due to yield concerns (Section 3). Another related work is a design space exploration of printed microprocessors [5]. However, the target technology was different (inkjet printed electrolyte-gated field effect transistors) and no microprocessors were fabricated.

While this work focuses on flexible microprocessors, it is important to note that such microprocessors may be part of a bigger IGZO TFT-based system. An example system may consist of flexible sensors and use near field communication implemented in flexible technology to perform IO operations with a flexible microprocessor. IGZO TFTs are already used in manufacturing flexible RFID tags consisting of antenna, modulator, rectifier, and RFID logic circuit [39, 75, 97]. IGZO electronics are also widely used in flexible sensors, including gas sensors [18, 89], pressure sensors [96], and in-situ biological sensors [8].

3 FLEXICORES: ARCHITECTURE AND MICROARCHITECTURE

ISA and microarchitecture design for FlexiCores depends both on the fabrication technology as well as the targeted applications.

3.1 Technology: Characteristics and Constraints

Our study focuses on metal-oxide TFT-based flexible microprocessors due to the well known cost and scalability advantages of such TFTs [67]. In particular, we explore design of 0.8 µm IGZO-based TFTs [56] using FlexLogIC, a state-of-art commercial 'fab-in-a-box' production line [79] (Figure 1). A large volume of commercial flexible integrated circuits has been manufactured using 0.8 µm IGZO TFTs [39, 97]. Several prior works on flexible processors [4, 11] also use this technology for manufacturing.

FlexLogIC supports IGZO circuits made using n-type TFTs with resistive pull-up. The circuits are manufactured on a 30 µm flexible polyimide substrate which enables the TFTs to flex up to a radius of curvature of 3 mm without damage [41]. To fabricate IGZO integrated circuits, FlexLogIC uses a proprietary manufacturing process that deposits layers of IGZO TFTs and resistors and four routable aurum-free metal layers onto a 200 mm wafer of polyimide which has been spin-coated onto a glass backing. After fabrication, the flexible polyimide wafer can be removed from the glass backing.

As with the nMOS silicon integrated circuits of the 1970s and 1980s, n-type logic-based 0.8 µm IGZO presents several challenges to computer architects. First, the devices have poor noise margin, high power consumption, and significant variation in V_{th} [4], all of which lead to yield and energy efficiency concerns. Second, the technology is several generations behind silicon-based CMOS in terms of size and speed [4], which leads to footprint concerns and limits applications to ones with relaxed performance requirements. Third, unlike silicon-based CMOS, nearly all power consumption

(>99%) in 0.8 µm IGZO is static power consumption [4]. This requires power reduction to be achieved primarily through area reduction.

Overall, the technology characteristics of 0.8 µm IGZO suggest that FlexiCores ISA and microarchitecture should be designed to reduce gate count to achieve low footprint and high yield, even if it is at the expense of performance. A low gate count implementation should also improve energy efficiency.

3.2 Applications: Expressiveness and Performance Requirements

The performance and energy characteristics of 0.8 µm IGZO (slow, high energy) suggest that the best applications for flexible processors are ones where the latency requirements are lax and the duty cycles are low. Also, yield concerns with 0.8 µm IGZO suggest that applications with low precision and data memory requirement would be a better fit than applications that require high precision or data memory (since support for higher precision and data memory in hardware usually requires higher gate count). Finally, the unique cost, thinness, and conformality advantages of 0.8 µm IGZO suggest that target applications must have cost, thinness, or conformality as one the primary requirements.

Table 1 lists a set of example applications that meet the above characteristics. Note that several of these applications are classifier applications; a large number of flexible applications may need to make classification decisions in the field. For example, a flexible smart bandage [65] may need to determine if a wound has healed. A flexible odor sensor on the package [85] may need to determine if milk has expired. A wearable, disposable skin patch can monitor blood pressure [46]. Several of the applications, such as Heart Beat Sensor, Light Level Sensor, Food Temperature Sensor, Humidity Sensor, Body Temperature Sensor, and Pedometer process and then perform thresholding on an input stream. Point of Sale (POS) Computation and Smart Labels require the ability to efficiently look-up data stored in a simple database or other data structure, and may make use of pseudo-randomness (e.g., in a visually dynamic display). Error Detection Coding (EDC) improves reliability of wireless communication, and thus any flexible microprocessor which transmits or receives data wirelessly must be able to execute computationally inexpensive error detection encoding or decoding.

Applications in Table 1 are drastically different than those supported by traditional 8 bit microprocessors (e.g., video games, pre-emptive multiprocessing operating system, spreadsheet applications, etc). For these flexible applications, duty cycle is often measured in seconds or longer, which means most architectures can satisfy the application performance requirements, even 4-bit architectures. This also means that performance is important only in so far as improved performance leads to improved energy. For example, any performance optimization makes sense only if it improves performance more than it increases static power consumption (recall, nearly all power consumption in 0.8 µm IGZO is static power). Furthermore, performance can be easily traded off if it helps improve area, energy, or yield.

Another observation is the data precision requirements for these applications are limited, with many applications requiring < 8 bit of precision. As such, not only are conventional 32-bit microcontroller architectures (e.g., ARM, RISC-V, MIPS) an overkill, even 'tiny' 8-bit microcontrollers (e.g., PIC10F, ATtiny102) support wider datapaths than are needed for many applications.

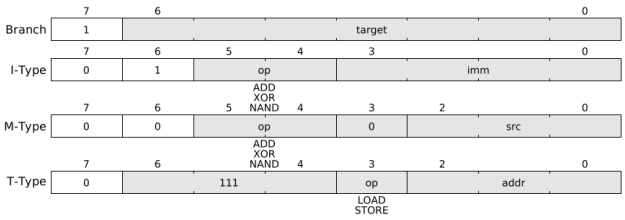
We use the above as guiding principles while designing ISA and microarchitecture for FlexiCore processors.

3.3 ISA

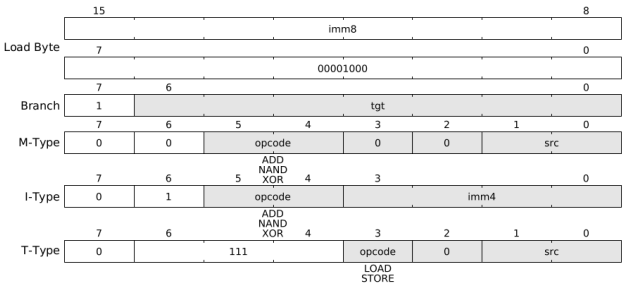
We decided to design a flexible microprocessor ISA with the following constraints, aimed at minimizing area (and thus increasing yield and decreasing energy), while sacrificing performance and expressiveness: the ISA must 1) support designs with < 800 NAND2 equivalent area, 2) have a ≤ 8 bit datapath, 3) support single-cycle instruction execution (since multi-cycle and pipelined designs may have energy and area implications), 4) have no more than 32 logic input and output ports (due to IO ring limitations in the current manufacturing process), 5) support integrated data memory (since external data memory may be cost-prohibitive for target applications).

We emphasize that our area constraint (< 800 NAND2 equivalent) is quite stringent. For reference, the area of the flip-flops in an ARMv6-m machine's register file is more than three times our allowable area. The on-chip data memory of the smallest PIC MCU (PIC10F200) is 704 NAND2 equivalent area, 88% of our area limit. The architectural registers of the Intel 4004, the first commercial microprocessor on an integrated circuit, have the area of 638 NAND gates, or nearly 80% of the area limit. Implementing only the register-file and memory of the subthreshold architecture of [71] would use more than 132% of the area limit. Since, like the PIC10F200 (and unlike the ARM or Intel machines), we assume limited access to external data memory, two things become clear. First, only applications which can be implemented with some small amount of finite memory can be supported. Second, data memory will be a significant fraction of die area, which further limits the combinational logic needed to implement ALU and control. As such, we look to an ISA which enables efficient instruction decoding and control word generation, and which has ALU operations which are inexpensive to implement.

Figure 2a shows FlexiCore4's (4-bit FlexiCore) ISA in terms of binary encoding of its instructions and a description of their semantics. FlexiCore4 supports a four-bit datapath (vs 32-bit datapath in [4] and 8-bit datapaths in [69]) since four bits is adequate for many target applications, while supporting our goal of high yield. FlexiCore4 ISA consists of a seven-bit program counter, a four-bit accumulator register, an eight-word (four octet) integrated data memory, and supports three arithmetic and logical functions (ADD, NAND, and XOR). We chose these functions deliberately since they support highly optimized ALU design (Section 3.4), while still supporting the logic and arithmetic primitives required by flexible processor applications. Since negation and conjunction are complete with respect to propositional logic, all two-input Boolean functions can be replicated with just the NAND and XOR functions only at the cost of four inverters. By using an accumulator architecture, we enable small instructions which allows an instruction to be fetched each cycle. An accumulator architecture also means only a single operand is stored in data memory, and thus only a single memory port is required. Data is transferred between the accumulator and memory using LOAD and STORE instructions. Control flow in FlexiCore4 is conditioned on the value of the



(a) FlexiCore4 ISA



(b) FlexiCore8 ISA

Figure 2: FlexiCore4's (a) nine instructions. The 'M-Type' instructions use the value stored in memory location 'addr' as the second operand, while the accumulator is both the first operand and the destination. The 'I-Type' instructions second operand is encoded directly in the 'imm4' field of the instruction. The 'Branch' instruction sets the program counter to the value stored in the 'imm7' field if and only if the most significant bit of the accumulator is set. The 'T-Type' instructions perform loads and stores — moving data between the accumulator and the memory address stored in 'addr'. FlexiCore8 (b) has all of the instructions of FlexiCore4, but with the addition of a 'Load Byte' instruction. This instruction loads an octet stored in the 'imm8' field into the accumulator.

accumulator, allowing conditional execution of basic blocks. This differs from the architecture in [69], which does not allow conditional execution of code. FlexiCore4 contains two four-bit IO buses – an input bus, and an output bus, used to asynchronously communicate with peripheral devices. These IO buses are memory mapped to addresses 0 and 1 of the internal data memory. The 8-bit instructions are delivered to the core asynchronously from an off-chip program memory. FlexiCore4's ISA is orthogonal — all ALU instructions may use either of the addressing modes: accumulator-immediate or accumulator-memory. The branch instruction jumps execution to an immediate program address conditioned on the most-significant bit of the accumulator. All instructions are encoded with a fixed eight-bit width.

In order to support applications with > 4 bit data requirements, we also designed an 8-bit ISA for FlexiCore8 - the 8-bit FlexiCore. The binary encoding of FlexiCore8s instructions is in Figure 2b. As with FlexiCore4, FlexiCore8 is designed to support a simple microarchitecture. Since doubling the size of the internal data-memory was area prohibitive (given the 800 NAND2-equivalent area restriction), we instead halved the number of words in memory, while doubling their width from a nibble to an octet. FlexiCore8 adds an additional LOAD BYTE instruction, which loads an 8-bit immediate value into the accumulator.

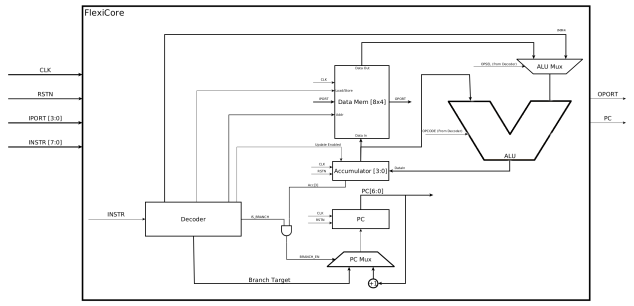
Since an important goal is to minimize logic area due to yield concerns with 0.8 µm IGZO, the ISAs reduce the complexity of the instruction decoders by embedding datapath control directly into the instructions themselves. For example, in FlexiCore4, instruction bits 5 and 4 are wired directly to the ALU output multiplexer select lines, and instruction bit 6 is wired directly into the input ALU Mux, selecting between immediate and memory based operands. Across both ISAs, only FlexiCore8's 'Load Byte' instruction is not encoded using 8 bit. This enables single cycle execution and removes the need for a complex state-machine based controller (a single flip-flop is used in FlexiCore8, and no state is used in FlexiCore4). These ISAs, then, are in stark contrast to earlier silicon-based commercial microprocessors with low device counts; their ISAs were optimized to enable compact programs instead. As such, these earlier microprocessors (e.g., MOS 6502, Intel 8080) often used variable length instruction encodings, and complex and stateful controllers to generate control words for each cycle of a multicycle execution. In the MOS 6502, for example, the 'ROM' portion of a controller produced 130 distinct 21-bit control words, depending on which stage of which instruction was being executed. Such an approach is infeasible for FlexiCores, which prioritize ultra low device count and high yield over compact programming.

Listing 1: Lack of expressivity leads to bloated code. This subroutine to perform a logical shift-right requires far more instructions than it would given a more expressive ISA.

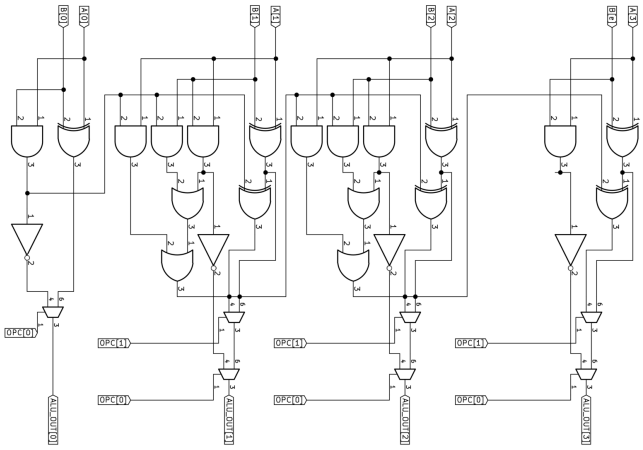
```
acc_right_shift:
    store r2; nandi 0; addi -3; store r4;
    nandi 0; store r5;
reverse_loop:
    load r2; br set_1; load r3; add r3;
    store r3; nandi 0; br loop_tail;
set_1:
    load r3; add r3; addi 1; store r3;
loop_tail
    load r4; addi 1; store r4; br left_shift;
    load r2; add r2; store r2; nandi 0;
    br reverse_loop;
left_shift:
    load r5; br done; load r3; add r3;
    store r2; load r5; addi 1; store r5
    nandi 0; br reverse_loop;
done:
    ...
```

The tradeoff of expressivity for architectural simplicity in the ISAs for FlexiCore4 and FlexiCore8 is not without drawbacks. Listing 1 shows how the FlexiCores' lack of expressivity can lead to a large number of static and dynamic instructions in program code. In this example, we see how performing a right-shift requires 36 static instructions and over 60 dynamic instructions. Although the cost

of higher number of dynamic instructions is somewhat offset by FlexiCore4's single-cycle execution (as opposed to tens of cycles for many CISC microprocessors), the high static instruction count leads to a requirement for larger program memory, which may offset the low cost of FlexiCores.



(a) FlexiCore Microarchitecture



(b) FlexiCore's ALU

Figure 3: FlexiCore4's microarchitecture. The ALU's logical implementation shows how the ALU's three functions can be built out of a single ripple-carry adder and a small number of additional gates.

3.4 Microarchitecture

FlexiCore4 is implemented as a single cycle machine (Figure 3) to minimize area. Also, microarchitectural decisions are made throughout the implementation to reduce gate count. For example, the ALU's ripple-carry adder performs bitwise logical exclusive disjunction and conjunction on the input words, meaning the ALU produces AND and XOR functions as side effects of addition, without requiring additional gate count. Similarly, we use the AND side effect to create the NAND function at the cost of only four inverters.

FlexiCore8's microarchitecture is similar to FlexiCore4's, with the required changes to datapath width and data memory. The only significant difference is that, upon identifying the LOAD BYTE opcode, FlexiCore8's decoder/controller sets a 'load byte' flag which indicates that the next byte fetched from program memory is not an instruction, but rather data to load into the accumulator.

A typical technique in traditional 8-bit microprocessors, which were often packaged in dual in-line packages with limited pin-counts, was to multiplex a single bus for both instructions and data (e.g., MOS 6502, Intel 8080). However, in order to achieve a single-cycle design (and thus not require additional flip-flops for registering partial instruction or data fetches), bus multiplexing is avoided in FlexiCores. Also, the IO ring in FlexiCores does not support inout buses. So, data multiplexing would have not been possible anyway.

We considered but ultimately rejected multicycle microarchitectures for FlexiCores (though we revisit the question later – Section 6). A multicycle microarchitecture can typically be used to minimize core area by temporally multiplexing hardware structures. However, due to FlexiCores' very simple ISAs, the amount of structure which can be reused is limited: the ALU's adder can be used to increment the program counter. Therefore, the area savings from this limited reuse are offset by the added control complexity (additional flip-flop, multiplexer, and control word generation), and such a multicycle microarchitecture would double the core's CPI (and hence double energy consumption).

3.5 Comparing FlexiCore processors

In this section, we present an analysis of area and power of FlexiCore processors and their components to quantitatively justify our ISA and microarchitectural design choices.

We designed a standard cell library of thirteen cells implemented in n-type logic with resistive pull-up (Figure 1) and using two of the four metal layers in the process. With this standard cell library, a standard EDA tool-flow can take a hardware design from SystemVerilog description to tape-out in GDSII, just as in conventional digital IC design. For physical design, we used an IO cell library with its own IO ring power supply (at 3 V and 4.5 V). Input IO cells have simple ESD protection to improve reliability and yield.

After synthesis and place and route, FlexiCore4 and FlexiCore8 have area of 5.56 mm^2 and 6.06 mm^2 respectively and static power of 1.8 mW and 2.4 mW respectively. For reference, we also synthesized a small, conventional silicon microcontroller, the TI MSP430, using the openMSP430 RTL [30]. In 0.8 µm IGZO, this core has an area of 170 mm^2, 30× larger than FlexiCore4. The MSP430 also consumes 41.2 mW static power, 23× more than FlexiCore4.

The results expectedly show that our choice of ISA and microarchitecture leads to much smaller core sizes than conventional simple microcontrollers such as MSP430. We also see that FlexiCore8 uses 9% more area than FlexiCore4, to support 8 bit datatypes. Section 4 discusses how this modest increase in area may impact yield.

Tables 2 and 3 show the area and power breakdown of the component modules of FlexiCore4 and FlexiCore8. Note that some core logic is not included in the submodules, and thus the sum of the module percentages does always add to 100%.

We observe that the on-chip data memory is the largest contributor to area and static power in both cores, with the on-chip data memory using the majority of design area and power in FlexiCore4. As this memory already uses a single read/write port, this suggests that

Table 2: Contribution of FlexiCore4 modules to overall core area and static power. The on-core, single-port data memory is the largest contributor to core area and power.

	ALU	Decoder	Regfile/Memory	PC	Acc.	Total Core
Area (% Non-Comb)	0%	0%	44%	27%	28.5%	36%
Area (% Comb)	100%	100%	55%	71%	71.5%	64%
Area (% of Core)	9%	1%	58.3%	23.4%	5.4%	100%
Static Power (% of Core)	7.9%	0.8%	57.5%	20.9%	5.8%	100%

Table 3: Contribution of FlexiCore8 modules to overall core area and static power. The on-core, single-port data memory is the largest contributor to core area and power.

	ALU	Decoder	Regfile/Memory	PC	Acc.	Total Core
Area (% Non-Comb)	0%	25.6%	41.5%	29%	71.5%	28.6%
Area (% Comb)	100%	74.4%	58.5%	71%	28.5%	71.4%
Area (% of Core)	15.5%	2.9%	40.9%	17.9%	10.8%	100%
Static Power (% of Core)	14.9%	2.7%	36.7%	17.4%	11.6%	100%

our choice of accumulator architecture for FlexiCores was a good decision since a choice of a different architecture (e.g., using load-store or memory-memory architecture instead of an accumulator architecture) would have required a second read-only access port. We estimated that adding a second port would have increased the data memory area by 39% and 25% for FlexiCore4 and FlexiCore8, respectively.

The relative sizes of the components differ between FlexiCore4 and FlexiCore8. In FlexiCore4, the combinational area of the data memory is larger than in FlexiCore8 in both relative and absolute terms. This is because the cost of the access port increases with the number of data words, and FlexiCore4 supports eight words, while FlexiCore8 supports only four words. Not surprisingly, FlexiCore8's ALU and accumulator are roughly twice as large as FlexiCore4's, due to 8 bit vs 4 bit datapaths. FlexiCore8 also has a larger controller, since FlexiCore8's load-byte instruction requires a more complex, stateful controller, while FlexiCore4's controller is stateless.

The high cost of data memory also suggests that architectures with support for small bitwidth datatypes are attractive. By supporting 4 bit datatypes, FlexiCore4's data memory can store twice as many words as FlexiCore8's. For the same reason, the decision to choose an ISA which can minimize decoder, ALU, and datapath area was important — all microarchitectural area (thus power) savings must come from these portions of the design for a given data memory size.

The low cost of the instruction decoder (< 3% area and power for both FlexiCores) suggests that a more complex decoder (and hence a more dense instruction encoding) may be a useful optimization for decreasing code size and reducing the number of IO pins required by the instruction memory bus.

Both FlexiCore4 and FlexiCore8 both have f_{max} of 12.5 kHz, which is considerably higher than the needs of most of our applications (Table 1). This further justifies our ISA and microarchitectural design decisions that trade off performance for lower gate count.

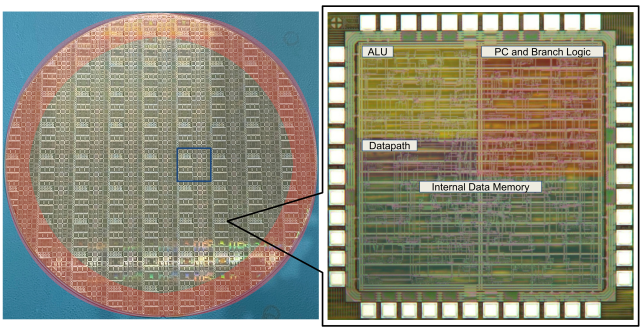
Instructions could be stored in an integrated program ROM whose contents are determined at tape-out (via mask ROM as in the flexible ARM processor [4]) or post-manufacturing (via write-once, read many laser programmable ROM). However, such an approach is inflexible as in either case the total program size must be known at tape-out, and a fixed program is insufficient for post-silicon testing. Instead, FlexiCores' programs are stored off-chip, and instructions are fetched via the dedicated instruction bus. This enables a single chip to support multiple applications, and allows arbitrary application sizes using an off-chip memory management unit (Section 5). It also enables rigorous testing of each manufactured chip, ensuring accurate yield measurements (Section 4). This approach is similar to that used in the flexible ALU work [69], which stored programs (and the program counter) on a separate flexible foil. A non-integrated program memory and off-chip memory management unit was a common technique used in low cost consumer electronics computing systems of the 1970s and 1980s. For example, the Nintendo Entertainment System's (NES's) CPU was a Ricoh 2A03 capable of addressing 64 KiB of off chip program and data memory, and the system itself contains a 2 KiB SRAM chip. However, as software often required both more data memory and significantly more than 64 KiB of address space to store program data, software was distributed in cartridges which contained ROMs for program data, additional SRAMs, and a 'memory management controller' (MMC) which facilitated the bank switching used to expand the default address space [15]. Banks were selected by writing to memory mapped registers in the MMC.

4 FLEXICORES: PHYSICAL IMPLEMENTATION AND ANALYSIS

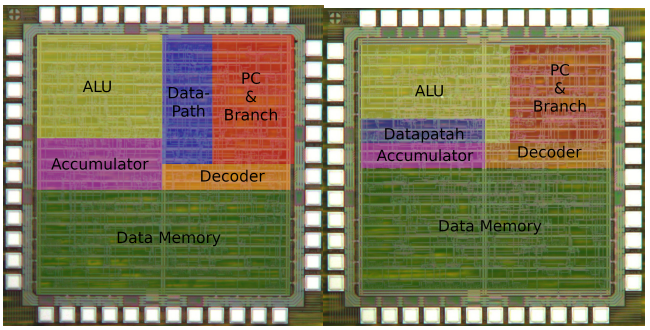
FlexiCore4 and FlexiCore8s were fabricated in 0.8 µm IGZO on multiple wafers with using FlexLogIC. Figure 4 shows a 200 mm biodegradable polyimide wafer containing 123 FlexiCore4 dies. The die photos shows the 336 gate FlexiCore4 and the 366 gate FlexiCore8. Each die is 9 mm^2 (including IO ring and pads), even though the FlexiCore4 and FlexiCore8 logic fit in areas of 5.56 mm^2 and 6.06 mm^2 respectively. FlexiCore4 uses 24 data pads and 8 power pads, while FlexiCore8 uses 31 data pads and 8 power pads. All power pads are duplicated. The die overlays show the relative sizes of core components. For FlexiCore8, the larger accumulator and datapath make up a large portion of the die area relative to FlexiCore4, whose area is dominated by data memory and the program counter. FlexiCore8 also has a considerably larger decoder (FlexiCore4's is less than 1% of die area and is not shown on the overlay).

4.1 Yield Analysis

Figure 5 shows the test-set up used to test the FlexiCore dies. After fabrication, the FlexiCore chips were tested on a semi-automated wafer probe station MPI TS2000 while still attached to a glass carrier. Yield is not significantly affected when chips are removed from the carrier. A test pattern derived from a Verilog simulation was translated to input signals generated by a NI PXIe-6570 Digital Pattern Instrument. Output signals were captured using the same instrument. All measurements were carried out at both 3 V and 4.5 V at clock frequencies up to 12.5 kHz. Due to limitations in the IO ring output buffers' ability to drive cable capacitance of the external test instruments, the circuits cannot be tested beyond 12.5 kHz. The FlexiCore dies were tested with over 100,000 cycles of random and directed test vectors. The test vectors stimulate all regions of the



(a) 200 mm **wafer with blow-up of a FlexiCore4 die.**



(b) Blow-up of a FlexiCore8 die **(c) Blow-up of a FlexiCore4+ (Section 6) die**

Figure 4: A 200 mm **polyimide wafer with FlexiCore4 dies, and blow-up photos of a FlexiCore4 die and a FlexiCore8 die. The red ring on the wafer indicates the** 16 mm **'exclusion' zone. Subfloat (c) is a die shot of FlexiCore4+, discussed in Section 6. Note that each chip has a different ratio of areas allocated to its components.**

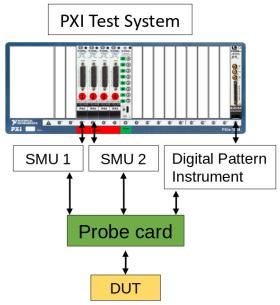
cores, with gates toggling on average 24,060 times, and all gates toggle at least once. The chips have been verified to work correctly more than six months after manufacturing, and can survive flexing to an 5 mm radius of curvature over 100 cycles.

Figure 6 shows the results of testing (in terms of number of observed errors) for FlexiCore dies at 3 V and 4.5 V for one randomly chosen wafer each for FlexiCore4 and FlexiCore8. We count a core as fully-functional if, at 3 V (4.5 V) supply voltage, a 1.5 V (2.25 V) threshold between logical LOW and HIGH, and 12.5 kHz clock rate, there are zero measured differences between its output and the expected output as determined by RTL simulation across all test vectors (i.e., the number of errors is zero). Table 5 shows yields for FlexiCore4 and FlexiCore8 at both voltages.

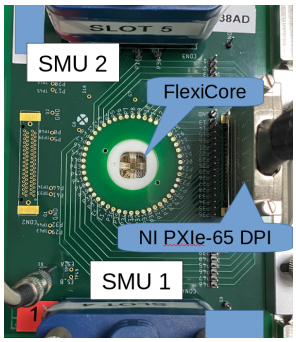
Ignoring the dies in the most external 16 mm of the wafer (due to edge effects), we achieved 81% yield for FlexiCore4 at 4.5 V, sufficient for sub-cent cost at high volume. This demonstrates the feasibility of building a flexible microprocessor at scale. Decreasing supply voltage from 4.5 V to 3 V degrades yield by 30%, likely due to timing faults. For FlexiCore8, the yield at 4.5 V is significantly lower (57%) than FlexiCore4's, likely due to the 9% higher gate count compared to FlexiCore4. This justifies our decision to target

Table 4: Comparison of different FlexiCores. Due to process refinement which occurred between manufacturing FlexiCore4 and FlexiCore8, FlexiCore8's power consumption is lower than FlexiCore4's. The process refinement involved increasing pull-up resistance by 50%. Applying the same refinement to FlexiCore4, we anticipate FlexiCore4 to consume 3.2 mW **on average.**

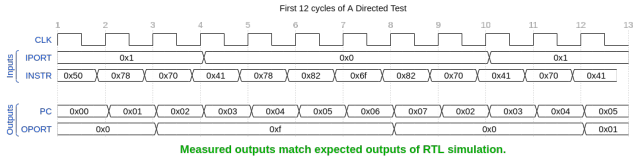
	FlexiCore4	FlexiCore8	FlexiCore4+
Area (mm^2)	5.56	6.05	6.4
Voltage (V)	4.5	4.5	4.5
Mean Power (mW)	4.9	3.9	3.4
Yield	81%	57%	n/a
Pin Count	25	31	24
Devices	2104	2335	2420
Clock Freq (kHz)	12.5	12.5	12.5
Datapath Width (bit)	4	8	4
Flexible	Yes	Yes	Yes



(a) Test set-up schematic



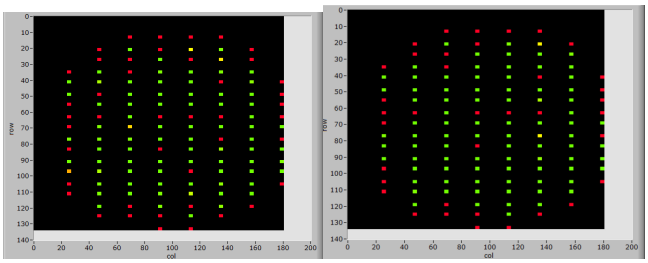
(b) A FlexiCore4 on glass carrier attached to probe-card.



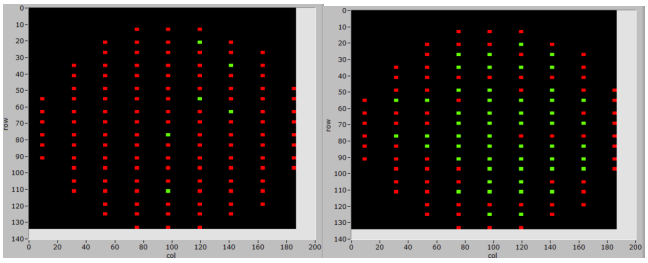
(c) Measured outputs of a working core

Figure 5: Schematic (a) and realization (b) of test set-up. Test vectors consist of over 100,000 cycles of directed and randomized tests. A FlexiCore die passes testing if and only if all measured results on all output ports match the expected output (c).

cores with physically small (< 800 NAND2 equivalent area) designs - recall that our ISA and microarchitectural decisions were made with this goal in mind. We also observe that decreasing supply voltage from 4.5 V to 3 V leads to a marked decrease in yield for FlexiCore8, rather than the modest decrease in yield experienced by FlexiCore4. This is likely due to FlexiCore8's 8-bit ripple-carry adder having a critical path twice as long as FlexiCore4's 4-bit adder.

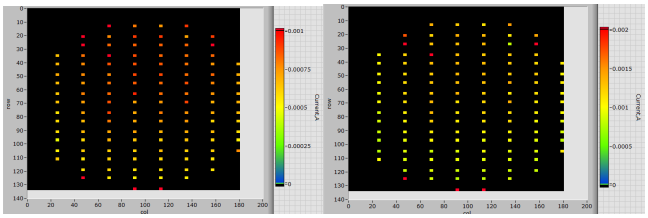


(a) FlexiCore4 at 3 V (b) FlexiCore4 at 4.5 V

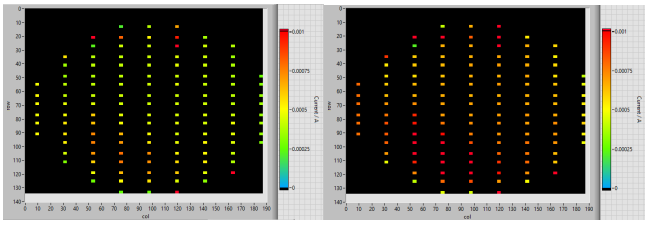


(c) FlexiCore8 at 3 V (d) FlexiCore8 at 4.5 V

Figure 6: Number of output errors on test vectors for FlexiCore cores at 3 V and 4.5 V. Green color means zero output errors and a functionally correct core.



(a) FlexiCore4 at 3 V (b) FlexiCore4 at 4.5 V



(c) FlexiCore8 at 3 V (d) FlexiCore8 at 4.5 V

Figure 7: Current draw on test vectors for FlexiCore cores at 3 V and 4.5 V.

The increased delay at 3 V prevents these circuits from meeting the 12.5 kHz timing requirement.

4.2 Process Variation

Process variation not only affects FlexiCores' yield (as discussed above), it also produces variability in power consumption. Figure 7 shows the current draw of FlexiCore4 and FlexiCore8 cores at 3 V and 4.5 V. Since previous work has focused on developing one-off prototypes of flexible microprocessors, this is the first report on the effects of process variation on flexible microprocessors.

Table 5: Yield for FlexiCore4 and FlexiCore8 at 3 V and 4.5 V. The table shows yield across the entire wafer, and yield after disregarding cores in the edge exclusion zone of the wafer.

	Full Wafer		Inclusion Zone Only	
	3 V	4.5 V	3 V	4.5 V
FlexiCore4	44%	63%	55%	81%
FlexiCore8	5%	42%	6%	57%

The average current draw for fully functional FlexiCore4 cores is 1.1 mA (0.73 mA) and current draw ranges from 0.8 mA to 1.4 mA (0.53 mA to 0.89 mA) on average during testing at 4.5 V (3 V). The average current draw for fully functional FlexiCore8 cores (on a different wafer - post a power reducing process update) is 0.75 mA (0.65 mA) and current draw ranges from 0.60 mA to 1.4 mA (0.36 mA to 0.42 mA) on average during testing at 4.5 V (3 V).

Both FlexiCore designs have large standard deviation in current draw at 4.5 V ($\approx$ 0.16 mA). However, at 3 V, the deviation is much more significant in FlexiCore4 than in FlexiCore8 (0.1 mA vs 0.03 mA). This is likely statistical noise due to the small number of FlexiCore8 cores which can operate at the target frequency at 3 V.

The high process variation can have significant impact on the number of usages of a flexible microprocessor given an energy budget. Techniques to reduce process variation and operate microprocessors efficiently in spite of it will be important.

4.3 FlexiCores in MOSFET CMOS

Given the low complexity of FlexiCores, their implementation in 5 nm process technology would allow hundreds of thousands of $\approx$ 0.03 mm$\times$0.03 mm FlexiCores per 300 mm silcon wafer. However, such small cores would be impractical to dice, with chips requiring 50 µm to 200 µm spacing using conventional diamond blades [102], wasting more than half to 90% of the wafer and increasing cost. Plasma dicing technology can reduce spacing to 10 µm, but is expensive [54]. Additionally, such a small die would be severely IO-limited, as each side will support 1-2 IOs at a 10 µm pitch, which is insufficient for a FlexiCore.

5 APPLICATION-LEVEL RESULTS

The field-reprogrammability of FlexiCore processors allows us to evaluate their performance and energy characteristics over a suite of representative kernels.

5.1 Representative Applications

Table 6 shows a list of benchmark kernels representative of computation in flexible computing applications (Table 1) that we use for our evaluations.

The 'calculator' application is a four function calculator which performs multiplication, division, addition, or subtraction of two inputs. Multiplication performs a 4 bit$\times$4 bit multiplication, producing an 8 bit output. Division produces the quotient and remainder of a 4 bit dividend and a 4 bit (non-zero) divisor. Addition (subtraction)

Table 6: Benchmark applications evaluated on FlexiCore. The 'interactive' four-function Calculator takes input operands and operations from the user before returning the desired computation. 'Reactive' applications interact with and run on the demand of a larger system. 'Streaming' applications perform computation on a stream of sensor inputs.

	Static Instructions	Application Type	Input Size
Calculator	352	Interactive	Operands + Operation
Four-tap FIR	177	Streaming	Per input
Decision Tree	210	Reactive	Depth 4, 3 features
IntAvg	132	Streaming	Per input
Thresholding	102	Streaming	Per input
Parity Check	105	Reactive	8-bit
XorShift8	186	Reactive	8-bit

generates a 4-bit sum (difference) with overflow (underflow). 'Decision Tree' performs inference on a randomly generated depth-four decision tree (such decision trees are suitable for several of the inference applications found in Table 1 [66]). 'IntAvg' performs exponential smoothing on an input sequence, perhaps generated by a sensor, generating a rolling, weighted average of the inputs as an output sequence. Exponential smoothing acts as a autoregressive infinite impulse response low-pass filter, and is a common mechanism for de-noising input sequences. As such, it may be used as pre-processing for thresholding applications such as Food Temperature Sensor, Light Level Sensor, and Humidity Sensor. Like, 'IntAvg', 'Four-tap FIR' benchmark applies a filter to an input stream, perhaps generated by a sensor, generating a filtered (low-pass, high-pass, or band-pass) output stream. Filter coefficients are in $\{-1, 1\}$. 'Thresholding' checks an input sequence for values greater than a specified threshold. The application places a non-zero value on the output bus if and only if the input sequence contains such an extreme value. 'Parity Check' computes the parity of an 8 bit word. Parity checking is a computationally inexpensive error detection code. 'XorShift8' [61] is a pseudo-random number generator which, given a non-zero seed, produces a length-255 sequence of non-repeating 8 bit numbers.

Programs are written in a highly readable assembly language. Programs are assembled into machine code binaries by a custom assembler written in Python. Some of the programs require more than 128 instructions (e.g., Calculator, Decision Tree). In order to execute these programs, we assume an off-chip memory management unit (MMU) which is driven by FlexiCore's output port. The MMU allows extending the program memory address space beyond the 128 instructions enabled by the 7 bit program counter alone. The MMU consists of finite-state transducer based controller, and a four-bit register. When the controller identifies a specific sequence of values on the FlexiCore's output port, it stores the value of the output port into the register after a short delay. This allows software to signal a 'page change' to one of sixteen different 128-instruction pages, and then branch to a desired location within that page. This technique can be extended to support arbitrary number of pages. This technique is non-standard and differs from traditional techniques using memory mapped IO, such as the NES's MMC. This nonstandard technique is used due to the IO constraints imposed by 1) lack of inout busses, and 2) desire to support single cycle and pipelined execution (versus Ricoh 2A03's CMOS 6502 multicycle core used in the Nintendo computer).

5.2 Performance and energy

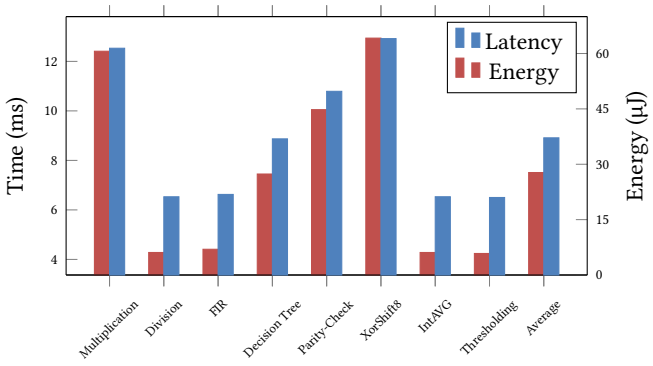


Figure 8: Performance and Energy for FlexiCore4 on benchmark kernels. From the calculator, we show the multiplication and division subroutines, as addition and subtraction are supported natively. Latency and energy includes time spent performing IO.

Previous works on flexible processors have relied on a simple set of test vectors or a single program to perform post-manufacturing validation. A working flexible microprocessor chip supporting complex or multiple applications has not been demonstrated by prior works.

Figure 8 shows measured energy and performance of FlexiCore4 on benchmark kernels. As the number of dynamic instructions is dependent on the input values, these values are based on the mean application latency given uniform sampling over the input space. When possible, we perform exhaustive tests across the input space. For Decision Tree we randomly sample from the input space, due to the large size of the input space. For the streaming kernels (IntAVG, Thresholding, FIR Filter), the latency and energy consumption is per input.

The latency of these kernels, on the order of ms, is acceptable given the duty cycles required of flexible computing applications (Table 1). However, energy consumption is high, and thus the performance of the FlexiCores may be constrained by energy rather than by performance limitations. For example, a FlexiCore4 which performs IIR filtering and then thresholding on an input stream in which data arrives once per second would consume 3.6 J d^{-1}, given perfect power gating. Using a commercially available 3 V, 5 mA h flexible battery [6], (and ignoring power consumption of program memory), this core can be powered for two weeks. This may be suitable for some applications, but other applications may need longer run-times; longer life-time may also decrease deployment cost [26]. Optimizations which reduce energy consumption may be needed to increase battery life and reduce system costs.

Table 7: Comparison of FlexiCore4 to other flexible ICs.

	This Work	PlasticARM [4]	Sharp Z80 [55]	UHF RFCPU [53]	8bit ALU [68]	ML IC [74]	Intel 4004
Device Count	2104	56,340	13,00	133,000	3504	3132	2250 Transistors
Area (mm^2)	5.6	59.2	169	93.45	225.6	5.6	12
Pin Count	28	28	40	N/A (RFID)	30	23	16
Voltage (V)	4.5	3	5	1.8	6.5	4.5	15
Power (mW)	4.05	21	15	0.81	Not Reported	7.2	1000
Clock Freq (kHz)	12.5	29	3000	1120	2.1	104	1000
Technology	0.8 µm IGZO-TFT	0.8 µm IGZO-TFT	3 µm c.g.-Si TFT	0.8 µm poly-Si TFT	5 µm organic + m-ox TFT	0.8 µm IGZO-TFT	10 µm
Logic Family	NMOS	NMOS	CMOS	CMOS	CMOS	NMOS	PMOS
NAND2 Equiv. Area	801	18,334	Not Reported	Not Reported	876	1024	N/A (Custom Logic)
Power Density ($mW\,mm^{-2}$)	0.723	0.355	0.0888	0.00867	Not Reported	1.29	83.3
Processor	Single-cycle accumulator ISA	Multicypcle (pipelined fetch) ArmV6-m	Multicycle (Z80 ISA)	Multicycle	8-bit ALU + Print Programmable ROM	Fixed-function ASIC	Multicycle I4004
Flexible	Yes	Yes	No (Corning 1737 Glass Substrate)	Yes	Yes	Yes	No
Programmability	Field Reprogrammable	At tape-out (integrated Mask ROM)	Field Reprogrammable	At tape-out (integrated Mask ROM)	Via seperate printed PROM and PC "foil"	None	Field Reprogrammable
Yield	81%	Not Reported	Not Reported	Not Reported	Not Reported	Not Reported	Commercial
Datapath Width (bit)	4	32	8	8	8	5	4

5.3 Comparison to Existing Flexible Processors

Prior flexible processors with prototypes are listed in Table 7. The fabricated 'PlasticARM' processor [4] uses a mask ROM for programs and is, therefore, not programmable. In any case, the ARMv6-m is significantly more expressive than the FlexiCores' ISAs. The resulting core, therefore, is an order of magnitude larger than FlexiCore4, and consumes > 5× the power of FlexiCore4. The Sharp Z80 is an implementation of the 8-bit Z80 ISA using TFTs. The fabricated processor has area ≈ 30× the size of FlexiCore4. The UHF RFCPU on a glass substrate is an inflexible prototype of a potentially flexible system that includes an antenna, an 8 bit CISC CPU, and 512 B of memory, built out of polysilicon TFTs. The CPU portion of this system is ≈ 3× the size of FlexiCore4. The '8bit-ALU' in [68] is built out of organic and metal-oxide TFTs and operates at f_{max} = 2.1 kHz. However, it can only support programs of 16 instructions, and its ISA does not support conditional branching, meaning it is incomparable to FlexiCores for most applications (e.g., decision tree, parity check, XorShift8, Calculator, FIR Filter). In addition to the cores in Table 7, FlexiCores greatly outperform the printed flexible processors described in [5, 69]. This is due to three orders of magnitude better switching performance of 0.8 µm IGZO than the organic and electrolyte-gated TFTs in those works.

6 OPTIMIZING FLEXIBLE PROCESSORS

Due to FlexiCore's emphasis on minimal core area, we traded ISA expressiveness for microarchitectural simplicity (Section 3). This trade-off enables commercially viable yield and sub-cent pricing for both FlexiCore4 and FlexiCore8, but it also increases programming difficulty and code size. Increased code size, in turn, can decrease energy efficiency and yield depending on the implementation, since storing the static code consumes power and area, and the high dynamic instruction count leads to increased program latency and energy. Therefore, it is important to find the right trade-off between core size and ISA expressiveness. This section considers the impact of supported operations and the number of operands per instruction, and microarchitecture on code size, core size, performance, and energy.

6.1 Revisiting operations in FlexiCore ISA

Listing 2: Unconditional branch with base FlexiCore ISA

```
    Branch  TGT
    Xori  0x8
    Branch  PRETGT
    ...
PRETGT:
    Xori  0x8
TGT:  ...
```

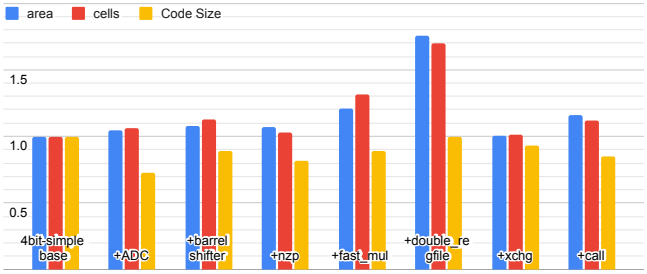


Figure 9: Changes in test code size and core area. With additional functionality, core size tends to increase while code size tends to decrease. Increasing the size of data-memory does not effect test code size, but it does allow for more complex programs which have larger data memory requirements.

We first explore the impact of increasing the expressiveness of the FlexiCore ISAs. Figure 9 shows the area, cell count, and code-size of FlexiCore4 with additional instructions or features enabled relative to the base design (Section 3). The features considered are a result of analysis of benchmark code – we identified 'code macros' and other small subroutine-like code sequences which are used repeatedly throughout the benchmark programs. Listing 2 shows how an unconditional branch which preserves the accumulator requires four static instructions in the base ISA. Three instructions are required to perform the branch, and an additional instruction is required ahead of the branch target to restore the accumulator to its original value. Data coalescing instructions (add with carry, subtract with borrow) enable addition and subtraction of integers which do not fit into 4-bits, and enable easily inspecting arithmetic overflow. Thus we consider adding ADC instruction. While left shifts are straightforward to perform in the base ISA through addition (and since the accumulator is 4-bits, the number of additions to implement left shift is never more than three), performing a *right* shift requires reversing the bit-order, performing a left shift, and then reversing the bit-order again. Thus, we consider a 4-bit barrel shifter which supports arithmetic and logical right shifts. Since branches in the FlexiCore4 are dependent only on the sign of the accumulator, many

control flow constructs are verbose (e.g., unconditional branches, which require multiple instructions). We consider enhancing the expressivity of branches by adding a three-bit branch mask which enables branching on whether the accumulator is negative, zero, or positive. We also consider adding a hardware multiplier, which can output either the top four or bottom four bits of a 4 bit × 4 bit multiplication. Finally, we consider doubling the size of the register-file. While this optimization does not affect code size, it enables running additional programs whose data does not fit into FlexiCore's limited memory. Figure 10 shows the impact of these ISA extensions on benchmarks.

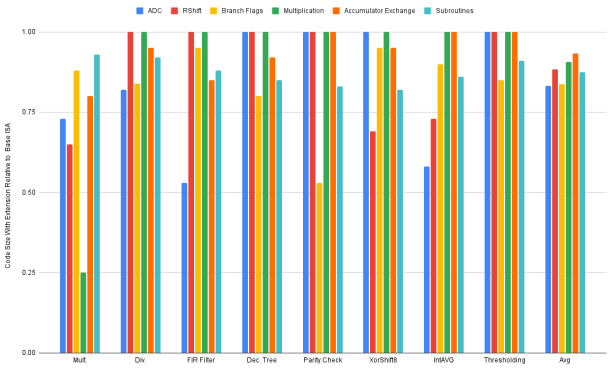


Figure 10: Code size of benchmarks with ISA extensions relative to baseline FlexiCore4 ISA.

The tradeoffs shown in Figure 9 allow us to revise the set of operations and features to be supported in a FlexiCore-like flexible processor. Due to the modest (< 10%) increase in area associated with the coalescing instructions, barrel shifter, and condition codes, we determine these instructions to be viable additions. Since the barrel shifter enables efficient computation of multiplication in software (or microcode), we decide to not add the hardware multiplier to the revised list considering the high gate count overhead for the multiplier. We also determine that the larger register file is not a viable change to the ISA due to its high (> 70%) area cost. We also, at the cost of 8 flip-flops, add a return address register, to the revised list enabling efficient subroutine calls. We also add an instruction that exchanges the values between the accumulator and a location in memory.

As such, the final set of operations to support are Add(i), Adc(i), Sub, Swb, And(i), Or(i), Xor(i), Neg, Xch (Mov(i) in load-store), Load, Store, Branch nzp, Call, Ret, Asr(i), Lsr(i).

We manufactured a small number of 4 bit, single-cycle accumulator machines with several of the ISA extensions (barrel shifter, branch condition flags). Figure 4c shows one of these 'FlexiCore4+' dies, which contains 15% more devices than FlexiCore4. At 4.5 V, these cores draw on average measured current of 0.75 mA, which is roughly the same as FlexiCore8.

6.2 Revisiting operands

While the above study shows that the ISA should support a number of additional features in order to increase code density, we recognize that overall energy and core area also depends upon the number of operands in the instructions, as well as microarchitectural features such as pipeline depth, and bus width. We performed a design space exploration in which we vary the number of operands (accumulator vs load-store), pipeline depth (single stage, dual stage, and multi-cycle), and instruction bus width. The cores are assumed to support the revised set of operations and operate at their SP&R f_{max}.

Figure 12 shows the normalized code size (in bits) and core area for the accumulator and load-store machines. Despite larger instructions, the load-store ISA has a slightly higher code density due to the extra expressivity of a second operand. However, the accumulator cores are smaller than the load-store cores. The single-cycle accumulator machine is the smallest design, as it requires neither pipeline nor controller state registers, and its datapath requires only a single data-memory port. Adding a second pipeline stage to the accumulator machine requires pipeline registers after decoding the fetched instruction. However, these additional registers are still less area than the additional area imposed by the single-cycle load-store architecture (i.e., a second data-memory port).

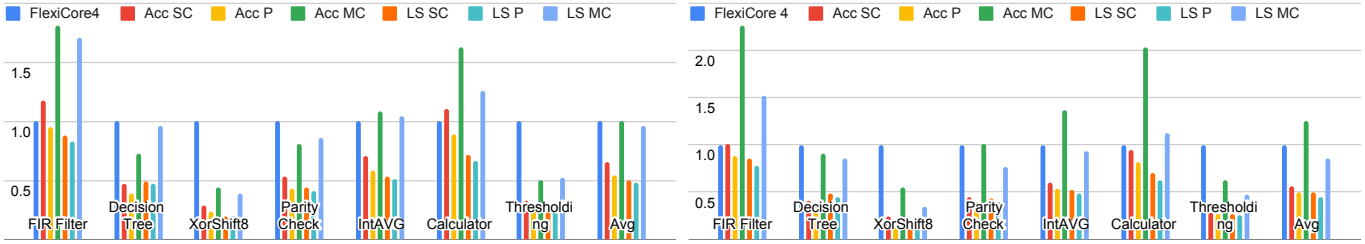
For the accumulator ISA, the multicycle design is the largest design, as not only does it require additional registers, it also requires generation of multiple sets of control words — one for each cycle of instruction execution. Since there is very limited opportunity for structure reuse (Section 3), overall area is higher than other accumulator-based designs. In the load-store architecture, the multicycle microarchitecture enables removal of the second register file port, and thus leads to an area savings substantial enough to offset the additional control complexity.

Figure 13 shows the energy consumption of the cores in two cases. First, we assume that the core has a program memory bus wide enough to fetch an entire instruction each cycle, as is the case in FlexiCore (Section 3). Under this assumption, the best performing core is the 2-stage load-store machine, which consumes less than half the energy of the baseline accumulator machine. However, when the program bus is restricted to the size of instructions used by FlexiCore, the load-store machine is incapable of fetching a new instruction each cycle, and thus the single cycle and 2-stage versions of the load-store machine are not possible. Hence, the best performing core is the 2-stage accumulator machine.

6.3 Application-level Results

Figure 11 shows the performance and energy of the new designs normalized against the baseline of FlexiCore4 for different benchmark kernels. Despite the 9-37% higher power consumption of the new cores (due to increased gate count to support new instructions as well pipelining), their improved performance leads to significant energy reduction (on average, the single-cycle and pipelined cores outperform FlexiCore4 by 53-115%, and consume only 45-56% of the energy of FlexiCore4), which was the primary goal of the ISA and microarchitecture optimizations.

Benefits depend on the kernel. Due to the increased sophistication of the branching instructions, the Decision Tree and Parity Check workloads, which feature a high percentage of branching instructions, see significant increase in performance. Both XorShift8 and IntAVG use right-shifts, and as such, we see significant improvements to



(a) Performance

(b) Energy

Figure 11: Performance and Energy Results of the DSE Cores on the Benchmark Applications normalized against FlexiCore4.

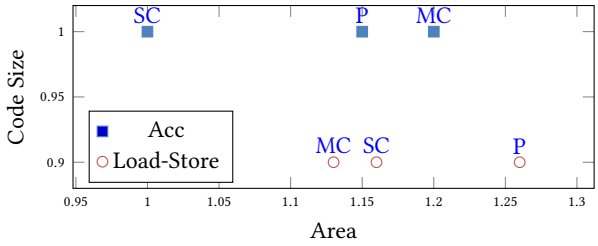


Figure 12: Normalized core area vs code size for accumulator and load-store machines with single-cycle (SC), two-stage pipelined (P), and multicycle (MC) microarchitectures.

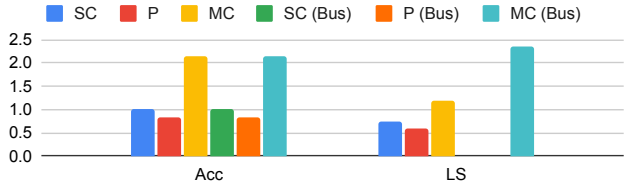


Figure 13: Relative energy consumption of the cores.

performance on these workloads, due to the difficulty of right-shifts on FlexiCore4 (Listing 1). The Calculator kernel sees large performance improvement on the load-store ISA, but only minimal improvement on the accumulator ISA. Although the data-coalescing instructions (ADC, SWB) do help to lower the dynamic (and static) instruction counts in the accumulator ISA, these savings are not significant on this kernel (recall, the kernel involves fetching operands and an operation from the user, doing a single computation, and then return the result, thus the ratio of I/O to computation is high). In thresholding, the increased expressiveness of the DSE ISAs leads to significant performance improvement. This is due to both the data-coalescing instructions and the improved branching instructions.

When a FlexiCore is built with an integrated program memory, the pipelined load-store architecture provides the best latency and energy characteristics, and is thus the preferred design point. When a FlexiCore is built without integrated program memory (necessitating IOs to fetch instructions), the pipelined accumulator design point is preferred, as its single-operand instructions require fewer IOs to fetch than the double-operand load-store instructions.

As static power consumption continues to increase relative to total power consumption in traditional electronics, the techniques used to minimize FlexiCore gate-count (e.g., using instructions as control word, choice of operations to minimize ALU complexity, accumulator-based ISA, etc) may become appealing even for ultra low power traditional electronics. In IGZO-TFT, a 95% reduction in device count led to a > 90% reduction in static power for FlexiCore4 compared to PlasticARM[4]. This ratio should be similar for electronics in traditional technologies. This could result in significant average power reduction for traditional electronics for low duty cycle designs, or for designs implemented in leading edge CMOS technologies - in both cases, static power is a significant fraction of overall power.

7 SUMMARY AND CONCLUSIONS

Natively flexible microprocessors have drawn a lot of recent interest as critical component to target applications whose computational needs cannot be met by traditional silicon-based electronics due to their conformality, thinness, or cost requirements. However, previous work has not shown that it is feasible to build flexible processors at scale (i.e., at high yield). Also, prior manufactured flexible systems were not field-reprogrammable and were evaluated either on a simple set of test vectors or a single program. A working flexible microprocessor chip supporting complex applications or multiple applications has not been demonstrated. Finally, no prior work performs a design space of flexible microprocessors to optimize area, code size, and energy of such microprocessors.

In this work, we fabricated and tested hundreds of FlexiCores - flexible 0.8 µm IGZO TFT-based field-reprogrammable 4 and 8-bit microprocessor chips optimized for low footprint and yield and showed for the first time that suitably-optimized flexible processors can have high yield (4-bit FlexiCores have 81% yield - sufficient to enable sub-cent cost if produced at volume). We evaluated these chips over a suite of representative kernels - the kernels take 4.28 ms to 12.9 ms and 21.0 µJ to 61.4 µJ for execution (at 360 nJ per instruction) - and measured process variation (relative standard deviation of 15.3% and 21.5% in terms of current draw of 4-bit and 8-bit FlexiCore chips respectively). These are first such studies for flexible microprocessors. Finally, we perform a design space exploration and identify design points much better than FlexiCores - the new cores consume only 45-56% the energy of the base design,

and have code size less than 30% of the base design, with an area overhead of 9-37%.

8 ACKNOWLEDGEMENTS

We thank the anonymous reviewers as well as the members of the Passat group for their feedback and the NSF for partial support.

REFERENCES

[1] Nathan Altice. 2015. *I am error: The Nintendo family computer/entertainment system platform.* MIT Press, One Broadway 12th Floor Cambridge, MA 02142.

[2] Márcio F Bergamini, André L Santos, Nelson R Stradiotto, and Maria Valnice B Zanoni. 2005. A disposable electrochemical sensor for the rapid determination of levodopa. *Journal of pharmaceutical and biomedical analysis* 39, 1-2 (2005), 54–59.

[3] Christopher J Bettinger. 2018. Recent advances in materials and flexible electronics for peripheral nerve interfaces. *Bioelectronic medicine* 4, 1 (2018), 1–10.

[4] John Biggs, James Myers, Jedrzej Kufel, Emre Ozer, Simon Craske, Antony Sou, Catherine Ramsdale, Ken Williamson, Richard Price, and Scott White. 2021. A natively flexible 32-bit Arm microprocessor. *Nature* 595, 7868 (2021), 532–536.

[5] Nathaniel Bleier, Muhammad Husnain Mubarik, Farhan Rasheed, Jasmin Aghassi-Hagmann, Mehdi B Tahoori, and Rakesh Kumar. 2020. Printed microprocessors. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, IEEE, Piscataway, NJ, 213–226.

[6] Blue Spark Technologies 2015. *UT Series Printed Batteries.* Blue Spark Technologies.

[7] RD Bringans and Janos Veres. 2016. Challenges and opportunities in flexible electronics. In *2016 IEEE International Electron Devices Meeting (IEDM)*. IEEE, IEEE, Piscataway, NJ, 6–4.

[8] Myung-Sic Chae, Ju Hyun Park, Hyun Woo Son, Kyo Seon Hwang, and Tae Geun Kim. 2018. IGZO-based electrolyte-gated field-effect transistor for in situ biological sensing platform. *Sensors and Actuators B: Chemical* 262 (2018), 876–883.

[9] Yu Chen, Guoqing Fu, Yael Zilberman, Weitong Ruan, Shideh Kabiri Ameri, Eric Miller, and Sameer Sonkusale. 2017. Disposable colorimetric geometric barcode sensor for food quality monitoring. In *2017 19th International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS)*. IEEE, IEEE, Piscataway, NJ, 1422–1424.

[10] Zheng Chen, John WF To, Chao Wang, Zhenda Lu, Nan Liu, Alex Chortos, Lijia Pan, Fei Wei, Yi Cui, and Zhenan Bao. 2014. A three-dimensionally interconnected carbon nanotube–conducting polymer hydrogel network for high-performance flexible battery electrodes. *Advanced Energy Materials* 4, 12 (2014), 1400207.

[11] Nitin Dahad. 2021. Flexible 6502 takes us back to the future. https://www.eetimes.com/flexible-6502-takes-us-back-to-the-future/

[12] Ravinder Dahiya, Nivasan Yogeswaran, Fengyuan Liu, Libu Manjakkal, Etienne Burdet, Vincent Hayward, and Henrik Jörntell. 2019. Large-area soft e-skin: The challenges beyond sensor designs. *Proc. IEEE* 107, 10 (2019), 2016–2033.

[13] Hiroki Dembo, Yoshiyuki Kurokawa, Takayuki Ikeda, Shusuke Iwata, Kazuaki Ohshima, Junko Ishii, Takuya Tsurume, Eiji Sugiyama, Daiki Yamada, Atsuo Isobe, et al. 2005. RFCPUs on glass and plastic substrates fabricated by TFT transfer technology. In *IEEE InternationalElectron Devices Meeting, 2005. IEDM Technical Digest.* IEEE, IEEE, Piscataway, NJ, 125–127.

[14] Hossein Derakhshandeh, Sara Saheb Kashaf, Fariba Aghabaglou, Ian O Ghanavati, and Ali Tamayol. 2018. Smart bandages: The future of wound care. *Trends in biotechnology* 36, 12 (2018), 1259–1274.

[15] Patrick Diskin. 2004. Nintendo Entertainment System Documentation. *Tokyo: Nintendo* 1 (2004), 1–47.

[16] Zihan Dong and Yuanwei Lin. 2020. Ultra-thin wafer technology and applications: A review. *Materials Science in Semiconductor Processing* 105 (2020), 104681.

[17] Ronald G Dreslinski, David Fick, Bharan Giridhar, Gyouho Kim, Sangwon Seo, Matthew Fojtik, Sudhir Satpathy, Yoonmyung Lee, Daeyeon Kim, Nurrachman Liu, et al. 2013. Centip3de: A 64-core, 3d stacked near-threshold system. *IEEE Micro* 33, 2 (2013), 8–16.

[18] Sunil Babu Eadi, Hyun-Jin Shin, P Senthil Kumar, Ki-Woo Song, R Yuvakkumar, and Hi-Deok Lee. 2021. Fluorine-implanted indium-gallium-zinc oxide (IGZO) chemiresistor sensor for high-response NO2 detection. *Chemosphere* 284 (2021), 131287.

[19] Kazuo Eguchi, Sujith Kuruvilla, Gbenga Ogedegbe, William Gerin, Joseph E Schwartz, and Thomas G Pickering. 2009. What is the optimal interval between successive home blood pressure readings using an automated oscillometric device? *Journal of Hypertension* 27, 6 (2009), 1172–1177. https://doi.org/10.1097/hjh.0b013e32832a6e39

[20] Rodger J Elble and James McNames. 2016. Using portable transducers to measure tremor severity. *Tremor and Other Hyperkinetic Movements* 6 (2016), 1–12.

[21] Tamer Elfaramawy, Cheikh Latyr Fall, Soodeh Arab, Martin Morissette, Francois Lellouche, and Benoit Gosselin. 2018. A wireless respiratory monitoring system using a wearable patch sensor network. *IEEE Sensors Journal* 19, 2 (2018), 650–657.

[22] enocean. 2017. Light Level Sensor – Ceiling Mounted. https://www.enocean.com/en/products/enocean_modules_902mhz/light-level-sensor-ells-oem/user-manual-pdf.

[23] Muhammad Fahad Farooqui and Atif Shamim. 2016. Low cost inkjet printed smart bandage for wireless monitoring of chronic wounds. *Scientific reports* 6 (2016), 28949.

[24] Muhammad Fahad Farooqui and Atif Shamim. 2016. Low Cost Inkjet Printed Smart Bandage for Wireless Monitoring of Chronic Wounds. *Scientific Reports* 6 (29 Jun 2016), 28949 EP –. https://doi.org/10.1038/srep28949 Article.

[25] DRAM Fastest. 1984. QL's Quest for business status. *Electronics & Power* 30, 11.12 (1984), 836–. https://doi.org/10.1049/ep.1984.0437

[26] Gregory P Forlenza, Taisa Kushner, Laurel H Messer, R Paul Wadwa, and Sriram Sankaranarayanan. 2019. Factory-calibrated continuous glucose monitoring: how and why it works, and the dangers of reuse beyond approved duration of wear. *Diabetes technology & therapeutics* 21, 4 (2019), 222–229.

[27] Wei Gao, Hiroki Ota, Daisuke Kiriya, Kuniharu Takei, and Ali Javey. 2019. Flexible electronics toward wearable sensing. *Accounts of chemical research* 52, 3 (2019), 523–533.

[28] Byron D Gates. 2009. Flexible electronics. *Science* 323, 5921 (2009), 1566–1567.

[29] David Tudor Gethin, Eifion Huw Jewell, and Tim Charles Claypole. 2013. Printed silver circuits for FMCG packaging. *Circuit World* 39 (2013), 188–194.

[30] O Girard. 2010. OpenMSP430 processor core, available at opencores. org.

[31] Shu Gong, Willem Schwalb, Yongwei Wang, Yi Chen, Yue Tang, Jye Si, Bijan Shirinzadeh, and Wenlong Cheng. 2014. A wearable and highly sensitive pressure sensor with ultrathin gold nanowires. *Nature communications* 5, 1 (2014), 1–8.

[32] Burton Grad. 2007. The creation and the demise of VisiCalc. *IEEE Annals of the History of Computing* 29, 3 (2007), 20–31.

[33] Giuliana Grimaldi and Mario Manto. 2010. Neurological tremor: sensors, signal processing and emerging applications. *Sensors (Basel, Switzerland)* 10, 2 (2010), 1399–1422. https://doi.org/10.3390/s100201399

[34] Pei He, JR Brent, Hui Ding, Jinxin Yang, DJ Lewis, Paul O'Brien, and Brian Derby. 2018. Fully printed high performance humidity sensors based on two-dimensional materials. *Nanoscale* 10, 12 (2018), 5599–5606.

[35] Gage Hills, Christian Lau, Andrew Wright, Samuel Fuller, Mindy D Bishop, Tathagata Srimani, Pritpal Kanhaiya, Rebecca Ho, Aya Amer, Yosi Stein, et al. 2019. Modern microprocessor built from complementary carbon nanotube transistors. *Nature* 572, 7771 (2019), 595–602.

[36] Holtek. 2018.

[37] Tsung-Ching Huang, Kenjiro Fukuda, Chun-Ming Lo, Yung-Hui Yeh, Tsuyoshi Sekitani, Takao Someya, and Kwang-Ting Cheng. 2010. Pseudo-CMOS: A design style for low-cost and robust flexible electronics. *IEEE Transactions on Electron Devices* 58, 1 (2010), 141–150.

[38] YongAn Huang, Hao Wu, Lin Xiao, Yongqing Duan, Hui Zhu, Jing Bian, Dong Ye, and Zhouping Yin. 2019. Assembly and applications of 3D conformal electronics on curvilinear surfaces. *Materials Horizons* 6, 4 (2019), 642–683.

[39] Ming-Hao Hung, Chung-Hung Chen, Yi-Cheng Lai, Kuan-Wen Tung, Wei-Ting Lin, Hsiu-Hua Wang, Feng-Jui Chan, Chun-Cheng Cheng, Chin-Tang Chuang, Yu-Sheng Huang, Cheng-Nan Yeh, Chu-Yu Liu, Jen-Pei Tseng, Min-Feng Chiang, and Yu-Chieh Lin. 2017. Ultra low voltage 1-V RFID tag implement in a-IGZO TFT technology on plastic. In *2017 IEEE International Conference on RFID (RFID)*. IEEE, Piscataway, NJ, 193–197. https://doi.org/10.1109/RFID.2017.7945608

[40] Texas Instruments. 2018. Nano-Power System Timer for Power Gating. http://www.ti.com/lit/ds/symlink/tpl5111.pdf.

[41] Hye-Won Jang, Gi-Heon Kim, and Sung-Min Yoon. 2020. Analysis of Mechanical and Electrical Origins of Degradations in Device Durability of Flexible InGaZnO Thin-Film Transistors. *ACS Applied Electronic Materials* 2, 7 (2020), 2113–2122.

[42] Nobuo Karaki, Takashi Nanmoto, Hiroaki Ebihara, Sumio Utsunomiya, Satoshi Inoue, and Tatsuya Shimoda. 2005. A flexible 8b asynchronous microprocessor based on low-temperature poly-silicon TFT technology. In *ISSCC. 2005 IEEE International Digest of Technical Papers. Solid-State Circuits Conference, 2005.* IEEE, IEEE, Piscataway, NJ, 272–598.

[43] Ulya R Karpuzcu, Abhishek Sinkar, Nam Sung Kim, and Josep Torrellas. 2013. Energysmart: Toward energy-efficient manycores for near-threshold computing. In *2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, IEEE, Piscataway, NJ, 542–553.

[44] W R Keatinge, M G Hayward, and N K Mciver. 1980. Hypothermia during saturation diving in the North Sea. *Bmj* 280, 6210 (Feb 1980), 291–291. https://doi.org/10.1136/bmj.280.6210.291

[45] Yasser Khan, Arno Thielens, Sifat Muin, Jonathan Ting, Carol Baumbauer, and Ana C Arias. 2020. A new frontier of printed electronics: flexible hybrid electronics. *Advanced Materials* 32, 15 (2020), 1905279.

[46] Joshua Kim, En-Fan Chou, Jamie Le, Sabrina Wong, Michael Chu, and Michelle Khine. 2019. Soft wearable pressure sensors for beat-to-beat blood pressure monitoring. *Advanced healthcare materials* 8, 13 (2019), 1900109.

[47] Jayoung Kim, William R. de Araujo, Izabela A. Samek, Amay J. Bandodkar, Wenzhao Jia, Barbara Brunetti, Thiago R.L.C. Paixão, and Joseph Wang. 2015. Wearable temporary tattoo sensor for real-time trace metal monitoring in human sweat. *Electrochemistry Communications* 51 (2015), 41–45. https://doi.org/10.1016/j.elecom.2014.11.024

[48] Jayoung Kim, Jeerapan Itthipon, Somayeh Imani, Thomas Cho, Amay Bandodkar, Stefano Cinti, Patrick Mercier, and Joseph Wang. 2016. Noninvasive Alcohol Monitoring Using a Wearable Tattoo-Based Iontophoretic-Biosensing System. *Acs Sensors* 1 (2016), 1011–1019.

[49] Myung-Gil Kim, Mercouri G Kanatzidis, Antonio Facchetti, and Tobin J Marks. 2011. Low-temperature fabrication of high-performance metal oxide thin-film electronics via combustion processing. *Nature materials* 10, 5 (2011), 382–388.

[50] Yong-Hoon Kim, Dae-Gyu Moon, and Jeong-In Han. 2004. Organic TFT array on a paper substrate. *IEEE Electron Device Letters* 25, 10 (2004), 702–704.

[51] Markus Krammer, James W Borchert, Andreas Petritz, Esther Karner-Petritz, Gerburg Schider, Barbara Stadlober, Hagen Klauk, and Karin Zojer. 2019. Critical evaluation of organic thin-film transistor models. *Crystals* 9, 2 (2019), 85.

[52] Lukas Kranz, Christina Gretener, Julian Perrenoud, Rafael Schmitt, Fabian Pianezzi, Fabio La Mattina, Patrick Blösch, Erik Cheah, Adrian Chirilă, Carolin M Fella, et al. 2013. Doping of polycrystalline CdTe for high-efficiency solar cells on flexible metal foil. *Nature communications* 4, 1 (2013), 1–7.

[53] Yoshiyuki Kurokawa, Takayuki Ikeda, Masami Endo, Hiroki Dembo, Daisuke Kawae, Takayuki Inoue, Munehiro Kozuma, Daisuke Ohgarane, Satoru Saito, Koji Dairiki, et al. 2008. UHF RFCPUs on flexible and glass substrates for secure RFID systems. *IEEE journal of solid-state circuits* 43, 1 (2008), 292–299.

[54] Christof Landesberger, Sabine Scherbaum, and Karlheinz Bock. 2011. Ultra-thin wafer fabrication through dicing-by-thinning. In *Ultra-thin Chip Technology and Applications*. Springer, 11 W. 42nd St Fl 15 New York, NY 10036, 33–43.

[55] Buyeol Lee, Yasuhiro Hirayama, Yasushi Kubota, Shigeki Imai, Akihiko Imaya, Mikio Katayama, Kiyoshi Kato, Akira Ishikawa, Takayuki Ikeda, Yoshiyuki Kurokawa, et al. 2003. A CPU on a glass substrate using CG-silicon TFTs. In *2003 IEEE International Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC.* IEEE, IEEE, Piscataway, NJ, 164–165.

[56] Jae Sang Lee, Seongpil Chang, Sang-Mo Koo, and Sang Yeol Lee. 2010. High-Performance a-IGZO TFT With ZrO2 Gate Dielectric Fabricated at Room Temperature. *IEEE electron device letters* 31, 3 (2010), 225–227.

[57] Sanghoon Lee, Qiongfeng Shi, and Chengkuo Lee. 2019. From flexible electronics technology in the era of IoT and artificial intelligence toward future implanted body sensor networks. *APL Materials* 7, 3 (2019), 031302.

[58] Jongwoo Lim, Hyunsung Jung, Changyeon Baek, Geon-Tae Hwang, Jungho Ryu, Daeho Yoon, Jibeom Yoo, Kwi-Il Park, and Jong Hee Kim. 2017. All-inkjet-printed flexible piezoelectric generator made of solvent evaporation assisted BaTiO3 hybrid material. *Nano Energy* 41 (2017), 337–343.

[59] Muhammad Hassan Malik, Giovanna Grosso, Hubert Zangl, Alfred Binder, and Ali Roshanghias. 2021. Flip Chip integration of ultra-thinned dies in low-cost flexible printed electronics; the effects of die thickness, encapsulation and conductive adhesives. *Microelectronics Reliability* 123 (2021), 114204.

[60] Giovanna Marrazza, Iva Chianella, and Marco Mascini. 1999. Disposable DNA electrochemical biosensors for environmental monitoring. *Analytica Chimica Acta* 387, 3 (1999), 297–307.

[61] George Marsaglia et al. 2003. Xorshift rngs. *Journal of Statistical Software* 8, 14 (2003), 1–6.

[62] Tilo Meister, Koichi Ishida, Reza Shabanpour, Bahman K Boroujeni, Corrado Carta, Frank Ellinger, Niko Münzenrieder, Luisa Petti, Giovanni A Salvatore, Gerhard Tröster, et al. 2015. Bendable energy-harvesting module with organic photovoltaic, rechargeable battery, and a-IGZO TFT charging electronics. In *2015 European Conference on Circuit Theory and Design (ECCTD)*. IEEE, IEEE, Piscataway, NJ, 1–4.

[63] microsensys GmbH. 2020. TELID 281.3Dm. https://ww.microsensys.ww.microsensys.de/fileadmin/user_upload/pdf-dateien/ds_sensor-transpond/TELID281M-03.pdf.

[64] Larry Mobley, Brian McMillin, and James Lewis. 2006. Vehicle ignition interlock systems having transdermal alcohol sensor.

[65] P. Mostafalu, W. Lenk, M. R. Dokmeci, B. Ziaie, A. Khademhosseini, and S. R. Sonkusale. 2015. Wireless Flexible Smart Bandage for Continuous Monitoring of Wound Oxygenation. *IEEE Transactions on Biomedical Circuits and Systems* 9, 5 (Oct 2015), 670–677. https://doi.org/10.1109/TBCAS.2015.2488582

[66] Muhammad Husnain Mubarik, Dennis D Weller, Nathaniel Bleier, Matthew Tomei, Jasmin Aghassi-Hagmann, Mehdi B Tahoori, and Rakesh Kumar. 2020. Printed machine learning classifiers. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, IEEE, Piscataway, NJ, 73–87.

[67] Kris Myny. 2018. The development of flexible integrated circuits based on thin-film transistors. *Nature electronics* 1, 1 (2018), 30–39.

[68] Kris Myny, Steve Smout, Maarten Rockelé, Ajay Bhoolokam, Tung Huei Ke, Soeren Steudel, Brian Cobb, Aashini Gulati, Francisco Gonzalez Rodriguez, Koji Obata, et al. 2014. A thin-film microprocessor with inkjet print-programmable memory. *Scientific reports* 4, 1 (2014), 1–6.

[69] Kris Myny, Erik Van Veenendaal, Gerwin H Gelinck, Jan Genoe, Wim Dehaene, and Paul Heremans. 2011. An 8-bit, 40-instructions-per-second organic microprocessor on plastic foil. *IEEE Journal of Solid-State Circuits* 47, 1 (2011), 284–291.

[70] L. Nazhandali, B. Zhai, A. Olson, A. Reeves, M. Minuth, R. Helfand, Sanjay Pant, T. Austin, and D. Blaauw. 2005. Energy optimization of subthreshold-voltage sensor network processors. In *32nd International Symposium on Computer Architecture (ISCA'05)*, Vol. 33. IEEE, Piscataway, NJ, 197–207. https://doi.org/10.1109/ISCA.2005.26

[71] Leyla Nazhandali, Bo Zhai, A Olson, Anna Reeves, Michael Minuth, Ryan Helfand, Sanjay Pant, Todd Austin, and David Blaauw. 2005. Energy optimization of subthreshold-voltage sensor network processors. In *32nd International Symposium on Computer Architecture (ISCA'05)*. IEEE, IEEE, Piscataway, NJ, 197–207.

[72] NXP. 2016. Intelligent MotionSensing Pedometer. https://www.nxp.com/docs/en/data-sheet/MMA9555L.pdf.

[73] Alvaro Ortiz Pérez, Vera Kallfaß-de Frenes, Alexander Filbert, Janosch Kneer, Benedikt Bierer, Pirmin Held, Philipp Klein, Jürgen Wöllenstein, Dirk Benyoucef, Sigrid Kallfaß, et al. 2017. Odor-sensing system to support social participation of people suffering from incontinence. *Sensors* 17, 1 (2017), 58.

[74] Emre Ozer, Jedrzej Kufel, James Myers, John Biggs, Gavin Brown, Anjit Rana, Antony Sou, Catherine Ramsdale, and Scott White. 2020. A hardwired machine learning processing engine fabricated with submicron metal-oxide thin-film transistors on a flexible substrate. *Nature Electronics* 3, 7 (2020), 419–425.

[75] Nikolas Papadopoulos, Firat Tankut, Babak Kazemi Esfeh, Marc Ameys, Florian De Roose, Bart Aerts, Steve Smout, Myriam Willegems, Raf Appeltans, and Kris Myny. 2021. 2cm diameter Antenna & Sharp Multi-threshold Detection Thin-film RFID Tags on Flexible substrate. In *2021 IEEE International Flexible Electronics Technology Conference (IFETC)*. IEEE, IEEE, Piscataway, NJ, 0059–0061.

[76] P Muditha Perera and Chamath Keppitiyagama. 2011. A performance comparison of hypervisors. In *2011 International Conference on Advances in ICT for Emerging Regions (ICTer)*. IEEE, IEEE, Piscataway, NJ, 120–120.

[77] Martin J Powell. 1989. The physics of amorphous-silicon thin-film transistors. *IEEE transactions on Electron Devices* 36, 12 (1989), 2753–2763.

[78] Alessandra Rinaldi, Claudia Becchimanzi, and Francesca Tosi. 2018. Wearable Devices and Smart Garments for Stress Management. In *Congress of the International Ergonomics Association*. Springer, Springer, 11 W. 42nd St Fl 15 New York, NY 10036, 898–907.

[79] Pragmatic Semiconductor. 2022. FlexLogIC Fab. https://www.pragmaticsemi.com/create-more/devices

[80] Bosch Sensortec. 2018. Digital humidity, pressure and temperature sensor. https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BME280-DS002.pdf.

[81] Bosch Sensortec. 2018. Digital Pressure Sensor. https://ae-bst.resource.bosch.com/media/_tech/media/application_notes/BST-BMP380-HS000.pdf.

[82] Food Safety Inspection Service. 1989. *USDA FSIS Directive 7110.3 Rev 1: Time/Temperature Guidelines for Cooling Heated Products.* USDA, 1400 Independence Ave., S.W. Washington, DC 20250.

[83] Anand Sharma, Nitesh K Chourasia, Vishwas Acharya, Nila Pal, Sajal Biring, Shun-Wei Liu, and Bhola N Pal. 2020. Ultra-low voltage metal oxide thin film transistor by low-temperature annealed solution processed LiAlO 2 gate dielectric. *Electronic Materials Letters* 16, 1 (2020), 22–34.

[84] LTD ShenZhen B.J.X. Industrial Development Co. 2015.

[85] Rajashekhar B Somasagar and Ashok Kusagur. 2017. Flavor Determination for Milk Quality Assessment using Embedded Electronic Noses. In *2017 2nd International Conference On Emerging Computation and Information Technologies (ICECIT)*. IEEE, IEEE, Piscataway, NJ, 1–4.

[86] Tomáš Syrovỳ, Robert Vik, Silvan Pretl, Lucie Syrová, Jiří Čengery, Aleš Hamáček, Lubomír Kubáč, and Ladislav Menšík. 2020. Fully printed disposable IoT soil moisture sensors for precision agriculture. *Chemosensors* 8, 4 (2020), 125.

[87] Toru Takayama, Yumiko Ohno, Yugo Goto, Asami Machida, Masashi Fujita, Junya Maruyama, Kiyoshi Kato, Jun Koyama, and Shumpei Yamazaki. 2004. A CPU on a plastic film substrate. In *Digest of Technical Papers. 2004 Symposium on VLSI Technology, 2004.* IEEE, IEEE, Piscataway, NJ, 230–231.

[88] Toshihiro Takeshita, Yusuke Takei, Takahiro Yamashita, Atsushi Oouchi, and Takeshi Kobayashi. 2020. Flexible substrate with floating island structure for mounting ultra-thin silicon chips. *Flexible and Printed Electronics* 5, 2 (2020), 025001.

[89] Hongyu Tang, Yutao Li, Robert Sokolovskij, Leandro Sacco, Hongze Zheng, Huaiyu Ye, Hongyu Yu, Xuejun Fan, He Tian, Tian-Ling Ren, et al. 2019. Ultra-high sensitive NO2 gas sensor based on tunable polarity transport in CVD-WS2/IGZO pN heterojunction. *ACS applied materials & interfaces* 11, 43 (2019), 40850–40859.

[90] Harold Thorgersen. 2000. Heartbeat monitor for wearing during exercise.
[91] Bhawna Tiwari, Pydi Ganga Bahubalindruni, Ana Santa, Jorge Martins, Priyanka Mittal, João Goes, Rodrigo Martins, Elvira Fortunato, and Pedro Barquinha. 2019. Oxide TFT rectifiers on flexible substrates operating at NFC frequency range. *IEEE Journal of the Electron Devices Society* 7 (2019), 329–334.
[92] Ge Tong, Zhou Jia, and Joseph Chang. 2018. Flexible hybrid electronics: review and challenges. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, IEEE, Piscataway, NJ, 1–5.
[93] Mani Teja Vijjapu, Sandeep Surya, Maruti Zalte, Saravanan Yuvaraja, Maryam Shojaei Baghini, and Khaled N Salama. 2021. Towards a low cost fully integrated IGZO TFT NO2 detection and quantification: A solution-processed approach. *Sensors and Actuators B: Chemical* 331 (2021), 129450.
[94] Stefan Wachter, Dmitry K Polyushkin, Ole Bethge, and Thomas Mueller. 2017. A microprocessor based on a two-dimensional semiconductor. *Nature communications* 8, 1 (2017), 1–6.
[95] Zifeng Wang, Funian Mo, Longtao Ma, Qi Yang, Guojin Liang, Zhuoxin Liu, Hongfei Li, Na Li, Haiyan Zhang, and Chunyi Zhi. 2018. Highly compressible cross-linked polyacrylamide hydrogel-enabled compressible Zn–MnO2 battery and a flexible battery–sensor system. *ACS applied materials & interfaces* 10, 51 (2018), 44527–44534.
[96] Chen Xin, Longlong Chen, Tongkuai Li, Zhihan Zhang, Tingting Zhao, Xifeng Li, and Jianhua Zhang. 2018. Highly sensitive flexible pressure sensor by the integration of microstructured PDMS film with a-IGZO TFTs. *IEEE Electron Device Letters* 39, 7 (2018), 1073–1076.
[97] Byung-Do Yang, Jae-Mun Oh, Hyeong-Ju Kang, Sang-Hee Park, Chi-Sun Hwang, Min Ki Ryu, and Jae-Eun Pi. 2013. A transparent logic circuit for RFID tag in a-IGZO TFT technology. *Etri Journal* 35, 4 (2013), 610–616.
[98] Yiran Yang and Wei Gao. 2019. Wearable and flexible electronics for continuous molecular monitoring. *Chemical Society Reviews* 48, 6 (2019), 1465–1491.
[99] Murat A Yokus, Cheyanne Hass, Talha Agcayazi, Alper Bozkurt, and Michael A Daniele. 2017. Towards a wearable perspiration sensor. In *2017 IEEE SENSORS*. IEEE, IEEE, Piscataway, NJ, 1–3.
[100] Xiaoqin Yu, Dan Liu, Lixing Kang, Yi Yang, Xiaopin Zhang, Qianjin Lv, Song Qiu, Hehua Jin, Qijun Song, Jin Zhang, et al. 2017. Recycling strategy for fabricating low-cost and high-performance carbon nanotube TFT devices. *ACS applied materials & interfaces* 9, 18 (2017), 15719–15726.
[101] Dalong Zhao, Devin A Mourey, and Thomas N Jackson. 2010. Fast flexible plastic substrate ZnO circuits. *IEEE Electron Device Letters* 31, 4 (2010), 323–325.
[102] Wang ZhiJie, Sonder Wang, JH Wang, Stephen Lee, Yao Su Ying, Richard Han, and YQ Su. 2005. 300mm low K wafer dicing saw study. In *2005 6th International Conference on Electronic Packaging Technology*. IEEE, IEEE, Piscataway, NJ, 262–268.