INVITED PAPER

Field-Programmable Wiring Systems

This paper discusses concepts of adaptive wiring manifolds, in which soft-configured wires can be formed "on demand" to form dedicated conductive pathways that can connect many points within an embedded system.

By Victor Murray, *Senior Member IEEE*, Marios Pattichis, *Senior Member IEEE*, Daniel Llamocca, *Senior Member IEEE*, and James Lyke, *Senior Member IEEE*

ABSTRACT | Field-programmable wiring systems refer to methods and hardware that can maintain the interconnection of components of different types. Generally, field-programmable wiring systems support the use of multidomain fabrics that can be used to route analog, power, digital signals, optical, microwave signals, etc. This paper reviews fundamental concepts associated with the practical implementation of field-programmable wiring systems. The paper also provides different implementation examples and discusses a list of challenges and recommendations for future work in this area.

KEYWORDS | Adaptive wiring panel (AWP); field-programmable wiring systems; reconfigurable manifold; self-healing circuits

I. INTRODUCTION

Field-programmable wiring systems are structures capable of forming programmable interconnections between the terminals of components that are attached to these structures, along with the methodologies and tools for managing these interconnections. Some type of programmable wiring concept will be at the heart of most reconfigurable system approaches. The termini of black boxes, fixed and configurable, that comprise such systems, must be interconnected, and programmable wiring approaches provide additional flexibility and can enhance the expressive capacity of components that would otherwise be united with a manifold of fixed connections. In fact, one methodology for creating a reconfigurable system involves flexibly connecting an arrangement of fixed components. In general, the interconnections making up systems can be fixed, programmable, dynamically reconfigurable, or based on a combination of these concepts.

The most well-known example of a programmable wiring system is the field-programmable gate array (FPGA) [1], [2]. FPGAs require programmable wiring to connect together configurable logic elements, internal memories, and other internally embedded intellectual property (IP) blocks in support of an overall personalization. In the 1990s, some dedicated field-programmable interconnect devices (FPIDs) were introduced into the marketplace [3], [4], but they were eventually discontinued. These devices were similar to FPGAs in that they used similar mechanisms for switching, routing, and configuration management, but they only provided interconnection routing functionality (i.e., no logic or memory resources).

FPGAs and FPIDs have traditionally been designed around digital applications. It is, for example, straightforward to use an FPGA to connect together two digital devices, and a wide range of interface styles (serial, parallel), signaling standards, and physical layer types are supported. It is not, however, generally possible to connect a lightbulb to a battery using an FPGA to program a wiring path between them. Not only are the voltage and current levels incompatible, but sometimes the signals that are routed through an FPGA are destroyed, manipulated, and regenerated as part of the routing process, which is sometimes performed as a computation. Hence, using FPGAs for general-purpose wiring chores is impractical.

We believe the need for a more versatile adaptive wiring approach will become more acute in the future as a broader range of reconfigurable systems ideas are advanced. The

Manuscript received February 2, 2015; revised April 28, 2015; accepted May 8, 2015. Date of publication June 1, 2015; date of current version June 18, 2015.
V. Murray is with the Department of Electrical Engineering, Universidad de Ingenieria y Tecnologia, 2221 Lima, Peru, and also with the Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131 USA (e-mail: vmurray@ieee.org).
M. Pattichis is with the Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131 USA (e-mail: pattichi@unm.edu).
D. Llamocca is with the Electrical and Computer Engineering Department, Oakland University, Rochester, MI 48309 USA (e-mail: llamocca@oakland.edu).
J. Lyke is with the Space Vehicles Directorate, Air Force Research Laboratory, Albuquerque, NM 87117 USA (e-mail: spaceelectronics@kirtland.af.mil).
Digital Object Identifier: 10.1109/JPROC.2015.2432123

authors have been involved in several projects spanning nearly two decades in pursuit of effective programmable wiring technologies for general applications, to include both the switch mechanisms themselves as well as fabrics employing them. The work culminated in the development of several conceptual and demonstration systems, including most recently a cellular system involving a number of planar tiles. A 6×8 array was formed as a composable, extensible system for application to adaptive wiring. This work involves not just concepts for making a scalable programmable wiring fabric, but also in creating a distributed configuration management system capable of accommodating other advanced features to query and manage state of thousands of switches within the demonstration system. Some of our demonstration fabrics were based on the use of MEMS-based latching microrelays (as well as volatile relays). The nonvolatile switches are of particular interest, since they permit wiring configurations to be set and maintained, even upon removal of primary power from the configuration management engine responsible for managing the wiring fabric. These early adaptive wiring manifold demonstrations were envisioned as being used as not only to replace fixed circuitry within electronic cards and boxes, but as a primary wiring medium (replacing wiring harnesses) for large-scale platforms, such as spacecraft.

Abstractly, a system with field-programmable wiring can be thought of as shown in Fig. 1. The system [Fig. 1(a)] contains a wiring medium ("adaptive wiring panel") with a number of sockets and discrete terminal pins. Devices (such as the "modules," also referred to as "black boxes") connect to the sockets (connectors) and pins present on the panel. In the programmable wiring system, the "cloud" is capable of manipulating a netlist (where each net specifies two or more termini that are to be interconnected) programmably, as suggested in the example in Fig. 1(b) using a "rat's nest" depiction. Different colors are shown to suggest that some nets may have different qualities from others, whether that is due to the nature of signals (e.g., power-bearing conductors may be differentiated from those containing sensitive analog signals or noisy digital switching waveforms) or some other distinguishing characteristic (cheap versus expensive, short versus long, fixed versus dynamic, etc.). While this paper is focused on electrical wire connections, the ideas described are applicable to other phenomenologies, such as optical [5] and fluidic [6] pathways.

Even at this preliminary level of discussion, it is possible to imagine some advantages to an adaptive wiring system. The first of these is the rapidity at which systems could be created. In principle, system configurations can be quickly formed if modules and blank (unprogrammed) panels are already in inventory by computing wiring configurations (as in FPGAs), aggregating the necessary components, assembling, and configuring the panel as necessary. By contrast, a custom fixed wiring system must be separately fabricated in a serial fashion, usually in a separate facility.

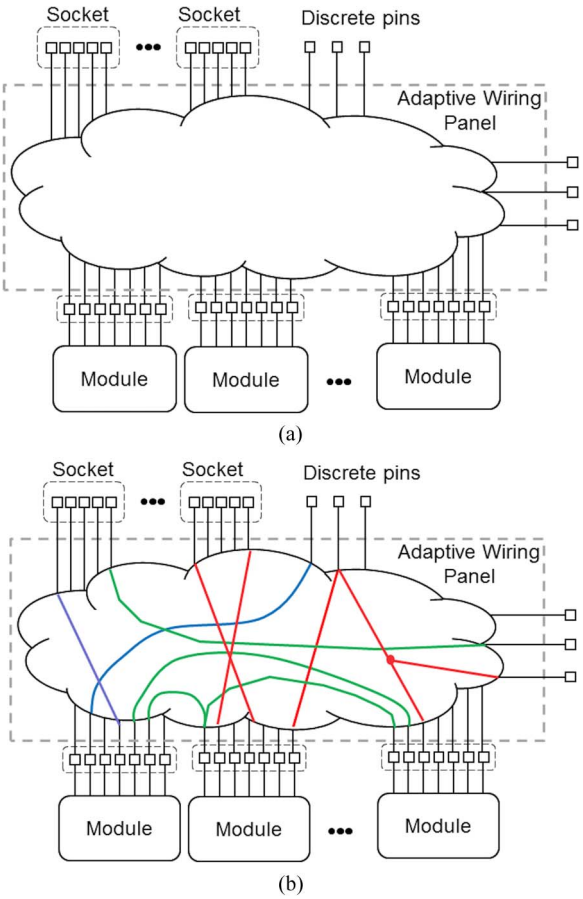


Fig. 1. ***Programmable wiring manifold. (a) Abstraction. (b) Depicting programmable interconnections.***

Since the panels are flexibly configurable, it is not necessary to manage an unbounded number of unique wiring panels, but rather a manageably small family of variants to address a wide variety of needs. We can illustrate additional prospective benefits. For example, as suggested in Fig. 2, it is conceivably possible using the dynamic nature of

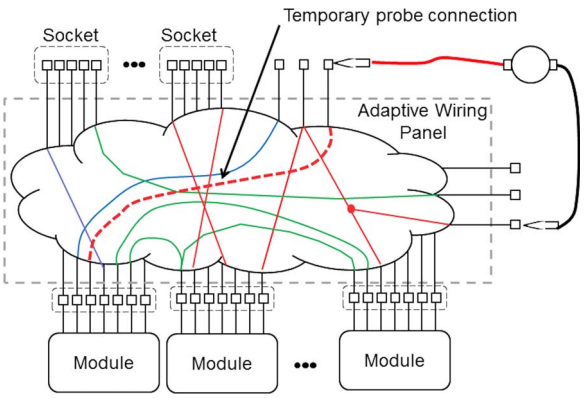


Fig. 2. ***Temporary connections in an adaptive wiring panel.***

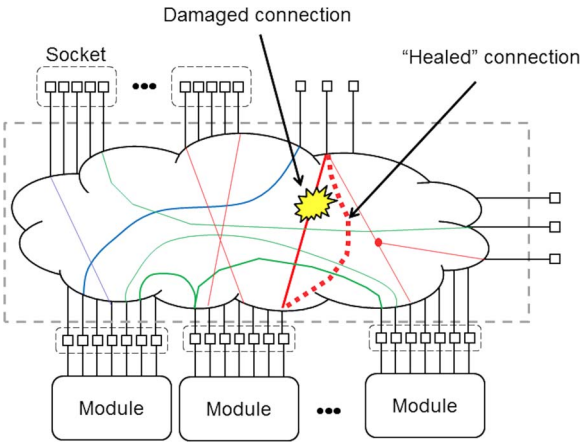


Fig. 3. *Repairing faulty connections in adaptive wiring system through circumlocution of damage.*

programmable wiring to form temporary probe connections on demand to isolate and connect to a specific terminal buried within a system. This feature can be valuable in system development and problem troubleshooting. Programmable wiring systems have an additional unique benefit, one in which it is sometimes possible to heal from faults occurring in the field. A fault is depicted abstractly in Fig. 3, which could be fatal to a fielded system. With programmable wiring, in some cases, it may be possible to recover from default by forming an alternate pathway.

It is also important to understand some of the inherent disadvantages and limitations in a programmable wiring systems approach. The first of these is in the addition of complexity and overhead. It is necessary to accommodate an infrastructure by which many switches are added to an otherwise passive wiring system, along with the configuration machinery necessary to manage configuration. Additional components can reduce reliability, especially if we take the pessimistic view that added components provide new opportunities for failure. Technically, we need to be concerned about the introduction of undesired parasitics and compromises in performance when comparing circuit traces and transmission line structures formed in custom design versus those equivalently provided in a programmable version of a wiring system. Unfortunately, the present implementations of the adaptive wiring concepts have been limited by technology, particularly in the limited availability of production quantities of compact, low power switches (especially nonvolatile) that are the principal foundation of an effective programmable wiring system. Many thousands of these switches are required to build even one instance of a simple wiring system. As a consequence, our prior demonstrations were of a more symbolic and academic (as opposed to practical) value.

Despite the present obstacles to the practical development of field-programmable wiring systems, we believe progress in technology will make them practical for a wider range of general applications, just as FPGA devices progressed from limited to broader applicability over time.

This paper is organized as follows. We begin with background information covering the basic concepts in Section II. We then cover practical considerations and discuss implementations in Section III. Recommendations for future work are given in Section VI. We provide concluding remarks in Section VII.

II. BACKGROUND

In this section, we consider fundamental elements that apply to field-programmable wiring systems. To make these ideas more precise, we introduce a mixed domain panel example. While our example depicts some implementation-specific concepts, it also illuminates some principles basic to any programmable wiring approach. We then discuss elemental concepts, including switches, domains, panel approaches, and basic graph theoretic principles useful for this work.

A. Basic Concepts

The first example [Fig. 4(a)] we consider is a simple programmable wiring panel (enclosed by the dotted line boundary), based on wires and switches that involves eight external terminals (A through H). The system also contains several interior termini (J through M), which are not directly accessible outside the boundary and would not be part of the external definition of panel interface. Switches are represented by circles, hollow meaning open and filled meaning closed. A pattern of closed switches defines a particular wiring configuration. As shown, a single connection between two external terminals (B and H) is formed by closing four switches. The connection of two or more terminals is referred to as a "net," and a collection of nets, each electrically isolated from the other, is referred to as a "netlist."

1) Graph Theory, Algorithms, and Applicability: A well-known graph-theoretic representation exists for routing

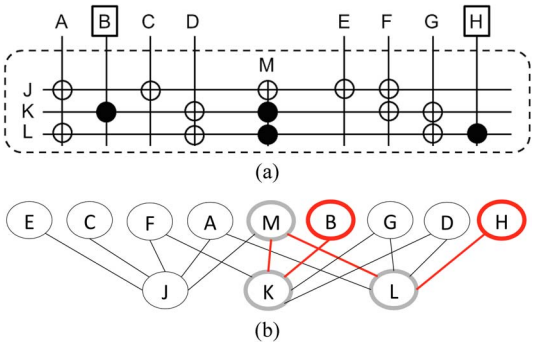


Fig. 4. *Very simple programmable panel. (a) Routing resource model of wires and switches. (b) Equivalent graph-theoretic formulation.*

problems involving switches and wires [7], and the equivalent graph for this example is shown in Fig. 4(a). Wires (including terminals) are modeled as nodes, and switches are modeled as edges. By convention, nodes corresponding to external termini are sometimes referred to as terminal nodes, whereas internal nodes (J, K, L, and M) are referred to as nonterminal nodes. Therefore, we have a graph $G = (V, E)$ where V denotes the set of vertices (nodes) and E denotes the set of edges (switches) between vertices. Refer to [8] for the notation. The graph-theoretic approach is ubiquitously applied in FPGA routing, and a rich body of literature exists extensively beyond the rudiments that we introduce in this section (see, for example, [9]).

The set of closed switches that implement a particular wiring configuration (implementing a netlist) represents a marking, defining a particular subgraph of the graph corresponding to a wiring system. The simplest concept in graph routing is connecting two nodes, which involves finding a marking (subgraph) for a path between these nodes. A walk from vertex v_1 to vertex v_k refers to a sequence of vertices $v_1, v_2, \ldots, v_k$ that results from following the edges $[v_1, v_2, \ldots, v_k]$. We have a closed walk when we return back to the starting vertex $v_1 = v_k$. A path is defined as a walk that does not include repeated vertices. We define a graph to be connected provided that we can find a path between any two nodes within the graph. Routing is only possible if a path between nodes exists. When this condition can be satisfied, one of the simplest methods of finding the shortest path is attributed to an algorithm described by Dijkstra [10]. When a solution exists, Dijkstra's algorithm can achieve a running time of $O(|E| \log |V|)$, where $|E|$ refers to the number of edges in the connected graph. It is important to note that we are not always concerned with finding a shortest path, merely one that satisfies our routing criteria, since often constraints will block selecting nodes and edges that might otherwise be in an optimal solution.

We define a component to refer to set of nodes that are connected. The number of isolated components of a subgraph is identical to the number of nets in the netlist. In Fig. 4(b), one such marking is shown for a netlist consisting of a single net (connecting terminals B and H). Bolded nodes are part of the marked solution, including the two terminal nodes (B, H) and three nonterminal nodes (K, L, and M). In general, the solution need not be unique, as other markings can often be found that will result in satisfying a netlist specification. In a net having more than two terminals, we are concerned with finding a spanning tree, a graph marking where a single path exists between all desired nonterminal nodes, while not allowing any cycles (repeated vertices in any path). The solutions for finding a tree are somewhat different than simple two-terminal routing. While, in principle, one could simply apply Dijkstra's algorithm iteratively on pairs of nodes within multiterminal nets, this will not generally result in the most efficient solution.

For any given graph, a spanning tree defines a subgraph that interconnects all of the vertices in the graph. In our case, the interconnection problem will very rarely require that all of the vertices be interconnected. Instead, we are interested in interconnecting vertices that will form a subset of the set of all vertices of the graphs. In this case, the graph that interconnects the given vertices is called a Steiner tree. Furthermore, the Steiner tree may contain additional vertices that are not in the original requirements set. These additional vertices are called Steiner vertices. Here, we think of the minimum cost spanning tree problem as one of determining the optimal Steiner tree that covers all of the vertices in a connected graph. This classic problem can be solved in $O(|E| \log |V|)$ time using the Prim and Kruskal algorithms (see [11] for details). As before, we do not always require a minimum spanning tree, just one that satisfies the constraints of a possibly more complex overall routing problem.

The general problem of graph routing in programmable wiring systems involves routing a netlist containing multiple nets simultaneously, which is sometimes referred to as solving a graph Steiner forest problem [12] [i.e., finding the solution to a number of independent (nonconnected) sets of nodes simultaneously in the graph-equivalent representation of a programmable wiring network] [13]. Unfortunately, even the basic problem of computing a single Steiner tree has been shown to be NP-complete [14], [15]. Hence, any effective solution to even basic graph routing problems requires the use of heuristics, and a rich body of these have been developed over many years for FPGA routing problems [16].

Many other elements of graph theory (such as node and edge weightings and colorings) are directly applicable to programmable wiring problem formulations. Weightings, for example, can be used to capture resource cost, propagation delay, or other important properties. We can define an edge cost function $c : E \rightarrow \mathbb{R}^+$ that assigns a cost (weight) with the use of each edge. With weightings, the generic instances of graph problems become node-weighted versions in which minimum (or maximum, depending on circumstance) tree or forest weightings must be solved for. Colorings can be used to mark resources according to application domain. For example, perhaps "blue" nodes and edges correspond to digital wiring resources, where as "green" nodes might correspond to power routing resources. Particular nets within a netlist can be marked according to such domains, informing the graph routing algorithm to seek suitable n-colorings of a particular wiring graph ($n = 2$ in this case). The graphs for such wiring problems are usually undirectional. In other words, we are only concerned with connecting the vertices with each other. Directionality, leading to directed graph formulations, may also be found in wiring problems. For example, in cases where a line repeater is employed within a programmable wiring system, it may be necessary to transfer signals in only one direction (e.g., from input to output), and directed graphs can be used to express this concept.

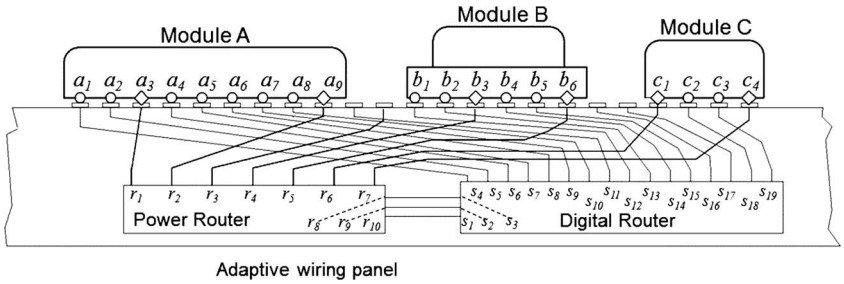


Fig. 5. ***Second adaptive wiring panel study example. In this case, only programmable connections are shown (i.e., all panel termini connect to active routers). On the other hand, the adaptive wiring panel can also contain a certain fraction of fixed connections.***

2) Multidomain Example: With these concepts in mind, next we consider a slightly more complicated example that demonstrates a few additional features, including some of the concepts just mentioned. This system (Fig. 5) consists of an adaptive wiring panel having three surface-mounted modules. The panel supports connections into different electrical domains, one being digital and the other being power. The figure illustrates some interior details, such as the use of active wiring routers dedicated to the particular electrical domains. Only programmable connections are shown (i.e., all panel termini connect to active routers), but an adaptive wiring panel can also contain a certain of fraction fixed connections. Each module contains both digital and power terminals, the latter denoted by the diamond shape (e.g., pins a_3 and a_9 in module A). In this particular panel design, termini belonging to the mounted modules correctly connect directly to corresponding pins belonging to the internal routers (e.g., digital pins connect to digital routers, power pins connect power routers). This would usually not occur by coincidence, but through enforcement of some standard convention relating to both panel and module layout. In other cases, some internal panel resources (other switching grids) might be involved in remapping the termini of connected modules so that they connect to routers capable of accommodating the appropriate signal type. Also, not every pin belonging to every module and not every panel pin is utilized, consistent with use cases in FPGAs (i.e., not every design uses every input–output pin).

Fig. 6 depicts a programmable resource model for the Fig. 5 example, indicating a set of external terminals and pinouts for the two routers. A number of terminals are also shown connecting between the two routers, suggesting the possibility that some resources can have mixed connections. Table 1 provides a sample netlist to be used as a problem example to map into this programmable wiring system, with examples of multiterminal nets and nets from different electrical domains.

One non-unique solution (also referred to as an embedding) of the Table 1 netlist in Fig. 6 resource model is shown in Fig. 7. Different colorings are used to label individual nets for improved clarity, and should not be

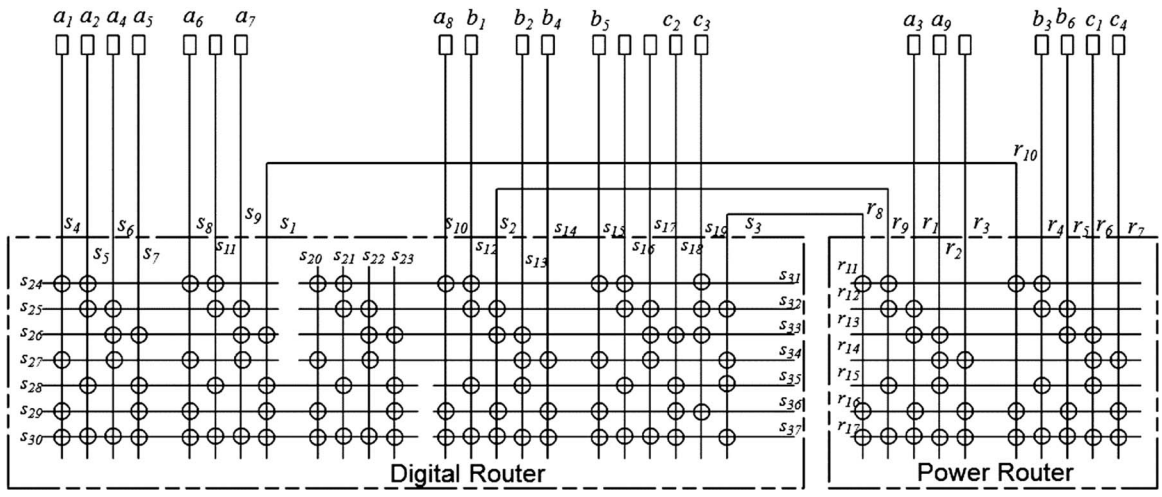


Fig. 6. ***Programmable resource model. We show an example with two routers, one digital and one power, and a set of external terminals and pinouts. The challenge is to connect different set of terminals using the two routers.***

Table 1 Netlist for Simple Example

Net	Pin connections	Connection type
1	a_1, b_1, c_2	digital signal
2	a_3, c_4	power connection
3	b_6,c_1	power connection
4	a_4, a_6, b_2,b_4	digital signal
5	a_2,a_5,c_3	digital signal
6	b_3,b_5	digital signal

confused with the colorings associated with graph theory discussed previously. This figure illustrates the overhead that can be associated with routing, as many switches and internal wiring resources are required to implement the desired netlist. In almost every case, only the wiring resources pertaining to a specific domain are used to implement solutions for particular net. The exception is in net 6, in which power pin (b_3) is deliberately connected to a digital pin in (b_5), possibly corresponding to a case where an input pin might be grounded in a user design. Such "domain crossings" must be carefully understood and planned for in the corresponding design and solution processes. The solution for this particular net involved routing through wires and switches in both digital and power routers. This very simple example demonstrates a congestion problem (indicated in Fig. 7) within the digital router in which three marked horizontal wiring resources are encircled (congestion). If it were necessary to add a seventh net (a_7 and a_8, shown by astericks in the figure), it would be impossible to connect them due to resource starvation. If it were possible to relocate the signal corresponding to a_8 to the terminal x, however it would be possible to connect pins together (for example, by closing switches on wire s_{25} that intersect columns s_9 and s_{11}, corresponding to pin $x = a_8$ and a_7, respectively). The notion of reassigning terminals to improve routability is sometimes referred to as addressing the "pin locking" problem [17]. Pin-locking is one of many examples of routing problems commonly occurring in FPGA systems that would also occur in more general field-programmable wiring systems.

B. Panel Considerations

1) Hierarchical Nature of Electronics Packaging: Interconnections are pervasively found to exist in systems as a hierarchical arrangement of structures, proceeding from the wires interconnecting transistors within an integrated circuit to the large-scale wiring harnesses in embedded platforms or the electrical conduits that make up buildings and other large-scale systems. If we consider the most fundamental wiring level as being the intracomponent wiring within a monolithic integrated circuit, such components (e.g., transistors) can be referred to level 0 (L0), then L0 wiring refers to the interconnection manifolds that make up an integrated circuit. It is here that we commonly find the programmable wiring systems inside FPGAs, themselves being a collection of L0 elements wired together to implement a programmable wiring function. The entire integrated circuit is considered to be a level I (L1) assembly, as are discrete components, as well as some hybrid circuits, multicomponent packages, and multichip modules. Hence, even L1 assemblies may exhibit a compound or recursive packaging structure. It is also conceivable that some of these elements may play the role of a field-programmable wiring system.

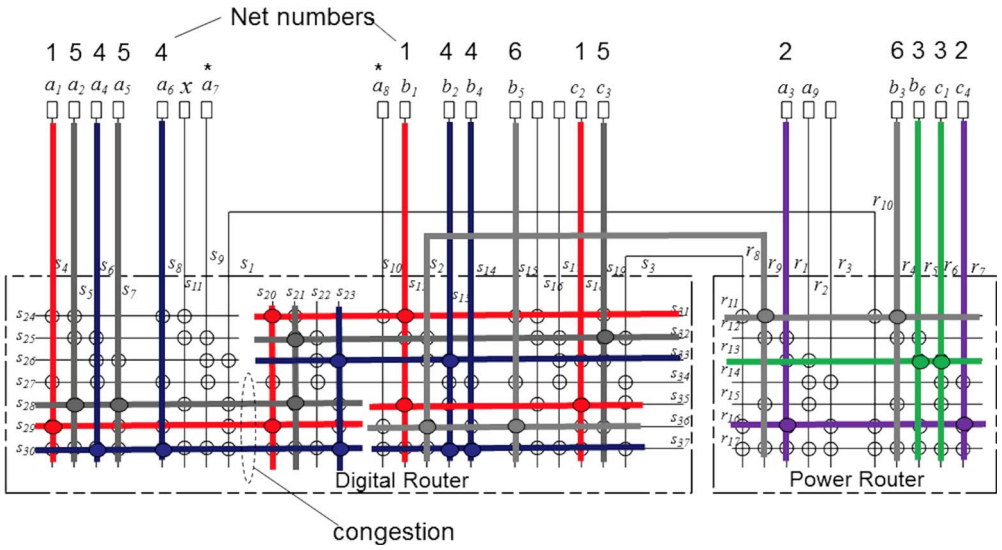


Fig. 7. *Sample implementation solution for multidomain netlist. Using the programmable resource model example from Fig. 6, different terminals are connected using the two routers and their terminals.*

For the most part, L1 assemblies do not stand alone, but must be combined to form higher level assemblies. This is most commonly done by arranging components onto printed wiring structures (such as rigid circuit boards or flexible wiring substrates). Circuitry combined in such a way can be referred to as a level 2 (L2) assembly. If some of the L1 components support programmable connections of their termini, these can be used, for example, to create a field-programmable circuit board. This was precisely the approach used in the 1990s by Aptix Corporation to make programmable systems in which a printed wiring board containing one or more field-programmable interconnect devices and prototyping regions for adding fixed components (integrated circuits and discretes) were used to form flexibly programmable assemblies to assist in prototype development.

Other levels of the present packaging hierarchy follow the same pattern. Circuit cards (L2 assemblies) can be aggregated, for example, through stacking or using a backplane, to form level 3 (L3) assemblies, sometimes called "boxes" (especially when inside a dedicated chassis). A number of L3 assemblies can be connected using cabling harnesses to form a level 4 (L4) assembly, and so on. This hierarchical notion is somewhat informal, and alternative forms of this overall recursive scheme are possible (e.g., the number of non-enclosed cards can be connected together with simple cables). In principle, most of these motifs lend themselves to exploration and treatment using a field-programmable wiring approach. While most of our discussions focus on the notion of programmable wiring panels, whether singly or in disjoint and/or tiled arrangements, it should be clear that this scheme can be applied at multiple levels. For example, a programmable printed wiring board (L2 assembly) could be considered panel that controls the connections between several mounted L1 assemblies. A large-scale embedded platform, such as a spacecraft, may employ an analogous concept in a "smart" panel (L4 assembly) to programmably connect a number of box-like (L3) assemblies.

2) Monolithic and Distributed Adaptive Panels: For the purposes of this paper, we will focus on (without loss of generality) a 2-D (planar) surface covered with electrical contacts belonging to one or more electrical domains. The contacts may be articulated into grouped regions (sockets) or distributed about the surface, as shown in Fig. 8. The termini are depicted in different colors, each color being a different signal domain (digital, high-speed digital, analog, power, microwave, and photonic). The distribution of terminal locations is uniform, but the population of contact types by color is not. Specifically, we show more digital domain contacts than contacts of other domains, reflecting an expectation that these types of signals dominate in wiring applications. Of course, the mixture and even the addition of new domain types can be considered. We have the modules represented as black boxes that need to be plugged into the wiring fabric, as depicted in Fig. 8(b). Although not shown, the bottom surface of these modules contains mating contact surfaces that align with the corresponding contacts on the panel substrate onto which they are placed.

We consider two broad classes of adaptive panels. The first of these is a monolithic panel or fabric. The second type is a distributed adaptive panel, especially a tileable or cellular adaptive wiring panel (a composite fabric), which is suggested in Fig. 9. In this case, rather than a large monolithic panel, a number of smaller panels are tiled together to form larger panels having different size, aspect,

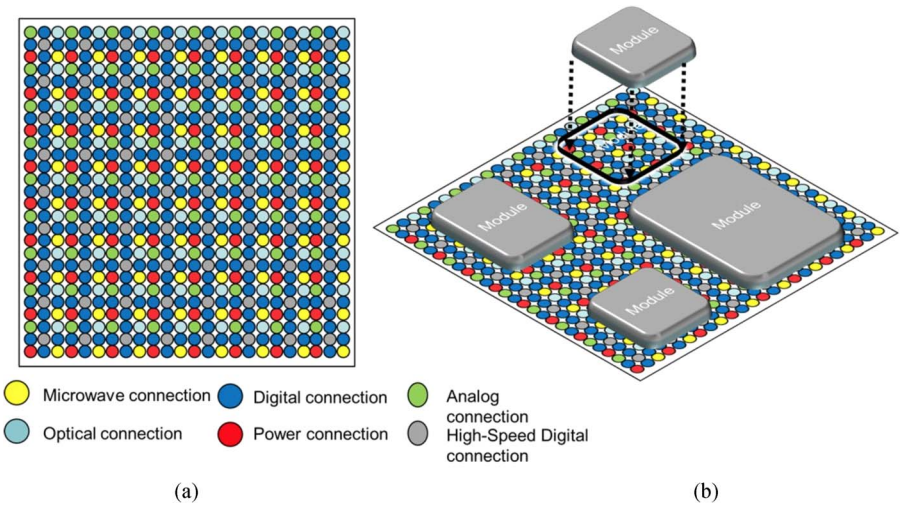


Fig. 8. ***Adaptive wiring panel (monolithic). (a) Panel depicting a distribution of multidomain contact points. (b) Depiction of modules placed onto the panel.***

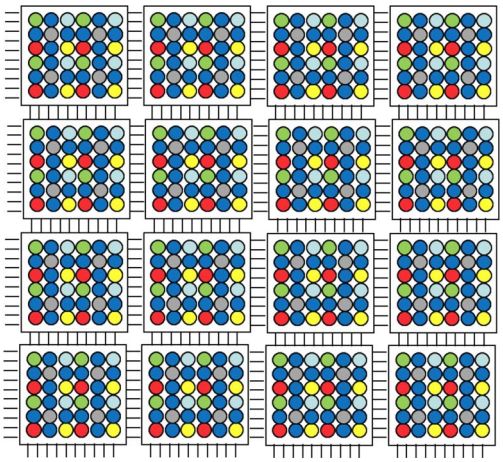


Fig. 9. *Tiled adaptive panel. This panel has been created using a number of small panels tiled together rather than using a large monolithic panel like the one from Fig. 8.*

and shape (essentially any legal shape that can be formed by an arrangement of tiles). In principle, the tiles, which can be thought of as unit cells, that do not need to be square as depicted here. Moreover, it is not necessary that the unit cells be planar (e.g., such as curved surfaces that might be formed from flexible or hinged tiles) or even confined to two dimensions. Cuboids and icosahedrons, for example, may also be considered, but our present discussions will be confined to planar tiles and surfaces. These cellular schemes are both enabled and limited by the number, type, and arrangement of intercell connections.

III. PRACTICAL CONSIDERATIONS

A. Application Domains and Regimes

A perfect programmable wiring system would form a set of persistent path connections between every desired point A and B, nonvolatile in nature (remaining intact until deliberately changed). Each connection would be electrically perfect, having zero resistance and inductance, no coupling or dispersion, capable of accommodating both large-signal and small-signal excursions, with zero time delay. In transmission line applications, the wires would have perfect guided wave properties and generate no discontinuities, reflections, or undesired phase shifts. Such a perfect wiring system can never exist outside of pedagogy, and we must thus consider compromises for real-world implementations. A possible approach is to consider a "divide-and-conquer" method in which we distinguish a number of electronic domains, for which a programmable wiring system can be optimized. This amounts to defining a number of coordinated subproblems, for which each is potentially handled with a different programmable wiring solution. This is the method suggested in Fig. 5, by using routers designed for different domains (digital and power). There is no magic dividing line for the number of domains necessary in a complex system. However, we can consider four reasonable partitions: digital, analog, power, and microwave. We summarize the ideal characteristics of each of these domains in Table 2.

The digital domain is concerned only with the transmission of discrete voltage values corresponding to Boolean states. Series loss and coupling are also important in the digital domain. Furthermore, signal integrity remains an important concern at a system level. Nevertheless, since logic levels are regenerated frequently in circuitry and most logic circuitry is designed with noise tolerance in mind, digital domain losses are more manageable than they are in other electronic domains. The primary interest in the digital domain is to achieve the highest possible density.

In the analog domain, signal qualities are far more important than in most designs. In series attenuation, losses due to coupling and dispersion can have severe impacts on signal qualities. The "analog" domain includes several subdomains. The small-signal subdomain consists of weaker signals (such as those from sensors) are manipulated through amplification, filtering, and other processes (including feedback control) where wiring quality can significantly impact performance. Density is usually less important, and (all other things being equal) wires in an analog design are traditionally higher in cross section (to reduce series loss) and spaced further apart (to reduce signal coupling). Attenuation through switches and meandering paths are among the many concerns in the analog domain. Analog power signals (usually amplification or high current switching) can be differentiated as a distinct subdomain, in which minimization of series loss is

Table 2 Qualities of Electronic Domains. We Use "tol" as an Abbreviation for Tolerance and "sens" for Sensitivity

	Electronic Domain			
Quality	**Digital**	**Analog**	**Microwave**	**DC Power**
Density	High	Medium	Medium	Low
Current handling	Very Low	Varies	Varies	High
Coupling sens.	Low	High	High	Low
Series loss	Tolerant	Low	Low	Lowest
Dispersion tol.	OK	Low	Very Low	OK
Timing tol.	Important	OK	Critical	OK

particularly important. The density for analog power connections is correspondingly lower than for small signal cases. Unless dynamic range is a significant concern, power analog signals can be more tolerant of coupling noise, but benefit from higher cross-sectional area.

The RF/microwave domain is possibly the most complex for field-programmable wiring systems. While at one level, RF/microwave signals can simply be considered as analog signals at much higher frequencies, they suffer from many other constraints owing to the temporal effects of managing guided electrical waves. Temporal shifts due to one length mismatch can critically affect performance, as can changes in characteristic impedance that can be brought on by practically any transition through switches, changes in geometry, differences in materials, all of which can represent discontinuities impacting circuit performance. The challenge of making a simple configuration change to move the signal from one wire to another can be far more involved in a microwave system. Thus, it is important to control impedances, account for phase length changes, and manage a complex set of discontinuities.

The management of direct-current (dc) power differs qualitatively in that series loss in wiring and switching that translates to degradations and power efficiency. Sometimes, increased capacitance in traces is actually desirable to maintain more stable power delivery. Cross-sectional area is usually high, and switch geometries are physically large.

B. Switches and Switching Mechanisms

Adaptive wiring systems depend on the use of switching mechanisms for implementing the interconnections. Historically, electromechanical relays provided the first effective switches. Relays, as physical (metal-to-metal) connections, principally allow routing of all electronic signal domains (e.g., see [18]). Unfortunately, they are physically large and expensive, and forming a manifold requiring thousands of connections would be very difficult. As such, the pursuit of effective relays at the microlevel or even nanolevel is of keen interest.

MEMS-based microrelays for one of the earliest identified uses of MEMS technology [19]–[21]. They were extensively studied, especially for radio-frequency (RF) applications [22]. In theory, such switches can be made thousands of times smaller than conventional electromagnetic relays. A number of actuation mechanisms have been used to make MEMS switches, including electrostatic [23], thermal [24], and electromagnetic [19]. Some approaches, the most attractive for applications to programmable wiring systems, involved bistatic mechanisms, which allow nonvolatile operation. Otherwise, the state of all switches in a system relaxes to a default state (usually open) upon removal of bias power. Such switches are less useful, as it is inconvenient for systems to effectively become unwired upon removal of power. As it turns out, to date, it is difficult to find production quantities of metal-to-metal latching MEMS relays.

Many MEMS relays are noncontacting, owing to the difficulty of forming a reliable contact surface, especially when switching live circuits, since arcing occurs further reducing lifetime and increasing contact resistance. For this reason, it is more common to see MEMS relays that do not have low dc resistance. Furthermore, the most common actuation mechanisms for MEMS relays has been electrostatic, which usually requires a higher dc voltage link into more complex circuitry for configuration management. For the most part, unfortunately the same switches are often volatile in nature. At the time of this writing, no adequate commercial solutions were available for developing adaptive wiring prototypes.

For these reasons, transistors remain the dominant switch mechanism for adaptive wiring applications, and they have been studied extensively for such applications in FPGA and FPID components. A transistor-based interconnection architecture for field programmability and testing was developed by Aptix Corporation [25]. More recently, FPGA architectures allow the interconnection of lookup tables (LUTs) with multiplexers to implement binary functions of several inputs. As an example, in [26], Llamocca and Pattichis described a dynamically reconfigurable architecture that demonstrated effective implementation of pixel processors as 8-b binary functions. Overall, FPGAs provide local routing mechanisms that are controlled by multiplexers and LUTs.

Beyond the use of wires for implementing connections, we could consider alternative methods for routing of light, heat, fluids, etc. Once again, MEMS technology has been applied to switching problems in these domains. However, we will not discuss these concepts further in this paper.

C. Connection "Quality of Service" (QoS)

It is useful to consider the concept of at least three possible "connection classes" within a programmable wiring system. In the first concept (the normal case in this work), a wire between point A and B, whether fixed or programmable, is traditionally considered to be a dedicated resource, available 100% of the time, even if it is never used in any electrically active way. This provides the highest QoS. It is possible to consider a more dynamical idea of timesharing, in which a wire that is unused for some amount of time can be repurposed to participate in other dedicated connections. This approach of "dynamic wires" comprises a second class of connection possible in some adaptive wiring approaches. If the schedule can be properly determined, the absence of such "just in time" wires during idle periods can seem transparent to the user. In other words, it has adequate QoS. This idea is embodied in the notion of time division multiplexing (TDM), which establishes a pattern (usually round-robin scheduling) of timeslots, each dedicated to a particular user/purpose for one or more wires. A similar concept of time slicing has also been discussed for FPGAs [27]. TDM buses are popular in digital communication networks, particularly high

reliability applications, due to the determinism associated with round-robin approach (using tightly controlled timing frames). They have also been criticized for inefficiency, despite their ability to displace the number of dedicated wires, due to the inability to efficiently accommodate variations in utilization across the different timeslots. Such considerations led to the evolution of more efficient (but less deterministic) packet-based networks (and their extensions in software-defined networking concepts), in which "content" is moved through the system even more dynamically, essentially in an on-demand fashion, both in terms of pathway and amount (i.e., bandwidth or duration of an effectively virtualized connection between source and destination). This third class of connection, which dominates the design of most networks in the world today, is qualitatively different than the other two classes, since the connection between source and destination does not take place in real time, but is subject to buffering delays that depend on the dynamic nature of traffic between all points in such systems. It is, therefore, not practical to use this third class of connection for connecting antennas to receivers, or the outputs of audio amplifiers to speakers, or even to battery connections. But the application of this third class a programmable connection is also not limited merely to data streams. They are, for example, inherent in the operation of some microfluidic concepts, which are concerned with the routing of droplets between multiple sources and destinations [28]. As such, we believe such concepts to have potential value in the design of adaptive wiring approaches.

D. Configuration Persistence and Latency

The methods by which programmable wiring systems can be configured as well as limitations of the switching technology dictate what sorts of applications these systems can be used for. Some programmable wiring concepts, for example, are only configurable once, such as antifuse FPGAs (an L1 concept) and at least one field-programmable multichip module (MCM) approach (an L2 concept) has been described [29]. Such approaches are useful for rapid construction and inventory reduction, but they cannot be repurposed after a system is deployed. Some switches have cycling limitations, they wear out after being switched a limited number of times. They are still useful in systems where switching is only episodic, such as periodic configuration changes. In some cases, it may be desirable to switch very rapidly, but the switch itself may be rate limited in switching speed. Electronic switches, of course, are much faster (submicrosecond) as compared to MEMS switches (milliseconds). Applications must consider both factors. A MEMS switch, for example, is not suitable for use as a packet router or in highly agile electronically steered arrays. They may be both too slow as well as endurance limited.

Ideally, configuration mechanisms would allow change but would maintain a nonvolatile state (e.g., support bistable operation). A configurable wiring harness used in systems designed in a traditional way might not fare well if the connections between primary batteries in critical subsystems suddenly disappeared upon shutdown. When connections much change dynamically, there is a concern of hot-switching effects (e.g., arcing) for some switch types, particularly those shunting larger amounts of power. Of course, system design methodologies could evolve to take volatility and hot switching into consideration, so that volatile wiring systems are systematically brought online in a carefully ordered way, when possible.

E. Empirical Design of Field-Programmable Wiring Architecture

Designing a programmable wiring system to first order is about choosing arrangements of wires and switches. While the goal of graph routing algorithms is to find solutions that allow a netlist to be embedded within a programmable fabric of these wires and switches, the goal of programmable wiring system design is to create good fabrics. By "good fabrics," we mean fabrics capable of expressing most user designs. Determining suitable fabrics is a nontrivial proposition. In general, it is neither possible to construct a brute force solution (capable of implementing all possibilities) nor is it possible to anticipate all user designs. The generation of benchmarks, both to test the effectiveness of programmable fabrics [30] (in this case FPGAs) and the tools that synthesize designs for them [31] is a long tradition in VLSI system development. It attempts to address these issues by creating a set of typical design use cases (the benchmarks) and testing them in candidate fabrics. The tests themselves may also be done with production or experimental algorithms and tools. Unfortunately, little work has been done to develop such benchmarks for programmable wiring systems, especially for multidomain designs that might mix of digital, analog, microwave, and power wiring requirements.

Brute force is not an option in fabric design. To recognize the challenges, note that full interconnection of n terminals requires the use of $n \times (n-1)/2$ switches. Thus, a brute force implementation of all possible connections leads to a quadratic growth in the number of required switches. Even for moderate numbers of terminals, the quadratic growth leads to prohibitively expensive implementations. This notion was understood from the earliest days of FPGA architectures [32]. Some elementary fabric concepts are shown in Fig. 10. A variety of heuristic strategies have been applied to counter the quadratic growth, such as fat tree, mesh of trees (MoT), and tree of meshes (ToM) implementations [33]. Most of the approaches replace a bruteforce crosspoint [Fig. 10(a)] with a segmented hierarchy of switch boxes [Fig. 10(b)]. The design of switchboxes also benefit from some principled design strategy. Neither a fully populated [Fig. 10(c)] nor a sparsely populated [Fig. 10(d)] are adequate [34], but rather some intermediate density of switches [Fig. 10(e)] chosen by some heuristic method [35], [36] are typically

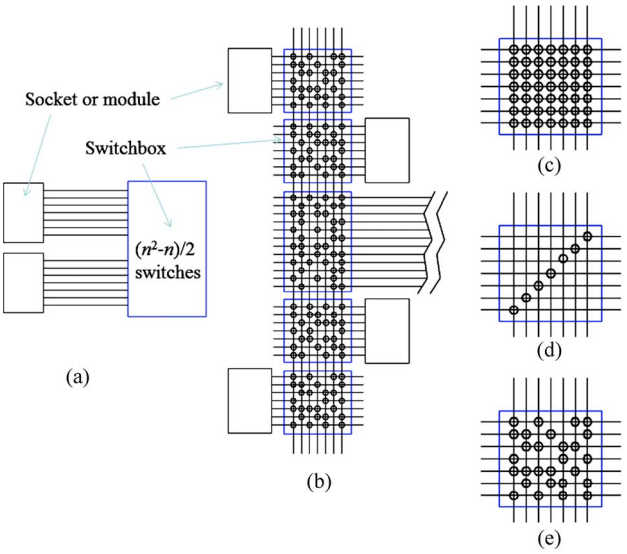


Fig. 10. *Elementary fabric concepts. (a) Brute force routing. (b) Segmented hierarchical fabric. (c) Fully populated switchbox. (d) Linear population. (e) Heuristically determined population.*

used in switchbox design [37]. While these design strategies provide a methodology for reducing the explosion of wiring resources, they do not address the question of heuristics in fabric design regarding what is "good enough" for practical purposes. It is largely a question of resource balance between wiring supply and wiring demand. The notion is straightforward: if the fabric supplies either too many or too few resources for most user designs, then there is an imbalance. In other words, fabric design is about supply and demand. If fabrics address the supply side of the equation, then benchmarks, as a collection of typical user designs, address the demand side.

The answer is not totally satisfying since benchmarks are empirical and must be curated. Over time, particularly when the complexity scale of typical systems grow, the benchmarks (if not refreshed) becomes stale, which was the experience of the FPGA industry [38]. It would be more satisfying if more formulaic guidelines existed, and some have indicated that empirical relationships, such as Rent's rule, can better inform the design of wiring architectures. Rent's rule is a power law relationship between the number of modules and exterior terminals in a system

$$T = k \cdot M^p \tag{1}$$

where T is the number of terminals, M is the number of modules, k is a constant, and $0 < p < 1$ is so-called Rent's exponent. The effect of Rent's rule [39] on architecture has been extensively discussed [40] for FPGA design.

The picture is further complicated in the more generalized wiring systems that include multidomain and multipanel fabrics. Generalized wiring architectures are multidomain fabrics, representing a mixture digital, power, analog, and microwave signals. As we have discussed, field-programmable wiring systems can be in a distributed form. These concepts represent enhancements over the basic FPGA routing concepts discussed so far. They are also amenable to graph-theoretic treatments. However, traditional FPGA routing algorithms would require extension to accommodate these additional features.

F. Multidomain Wiring

For each signal type, we also have a corresponding switch type. Thus, the implementation of multidomain fabrics requires the consideration of the distribution of different switch types. Typically, the fabric would include a small number of expensive power switches and a much larger number of digital switches. Beyond switch types, routing introduces additional issues that need to be dealt with. For example, analog and microwave routing requires that we deal with resistance build up, line length adjustment, and impedance tuning. Many of these could be represented using switchable circuits, such as binary resistive ladders to form a tunable resistance or selectable true-time delays.

G. Multipanel Wiring

In multipanel programmable wiring systems, each panel independently contains programmable wiring resources. In many respects, the challenges in developing multipanel programmable wiring systems are similar to those in multi-FPGA systems. In multi-FPGA systems (driven by large-scale hardware emulation applications [41], [42]), problems in partitioning large designs [43], performing pin assignment [44], and "spread-net" (nets cutting across two or more FPGAs) routing [45] have been studied. Just as there are non-unique solutions to the routing problem within a particular programmable wiring system, there are also non-unique solutions to the global partitioning of the wiring problem into subproblems. This sort of global/local problem at a large scale is analogous to that of the global/detailed routing problem for monolithic FPGA devices [46].

IV. SUMMARY OF EXPERIMENTAL DEVELOPMENTS IN FIELD-PROGRAMMABLE WIRING

In this section, we describe three generations of adaptive wiring research spanning two decades, sponsored by the Air Force Research Laboratory (AFRL) and one of its precursor organizations (Phillips Laboratory). The original impetus for early Phillips Laboratory investments in adaptive wiring research (mid-1990s) was the pursuit of fault-tolerant spacecraft design. The period was marked by large investments in MEMS technology development [dominated by the Defense Advanced Research Projects Agency

(DARPA)], and the laboratory studied promising MEMS-based latching switch designs employing thermal actuation and high aspect ratio (e.g., low resistance, high current handling) planar, metal-to-metal contact geometries. Since the switch designs supported nonvolatile reconfiguration and could be formed at relatively high densities (> 1000 switches/cm^2), the idea of creating smart wiring systems seemed a natural application. During that time, members of the AFRL research team already had experience with one of the Aptix prototyping systems [25], in which printed wiring boards employed several FPID devices and prototyping regions. Using this arrangement, components mounted to the prototyping regions could be interconnected by programming the FPIDs through a configuration interface.

The original concept for an exploratory adaptive wiring manifold based on MEMS would involve adapting the Aptix board concept, replacing the volatile FPID devices with more robust, nonvolatile versions based on MEMS switches. It was believed that such a system would be useful as fieldable wiring system, since the wiring patterns could remain formed once configured (with bistable nonvolatile switches), and the switches would be capable of sustaining wider voltage and frequency ranges than the complementary metal–oxide–semiconductor (CMOS)-based switches present in the Aptix FPIDs. Such a concept could be adapted to replace traditional boards and wiring harnesses, just as FPGAs had begun to find their way into an increasing number of production systems. These early sentiments provide impetus for a body of work spanning nearly two decades, described in this section.

A. Initial Conceptual Definition

It was envisioned that the original adaptive wiring system would be based on several implementation principles.

1) Creation of a relatively dense MEMS switch array with the configuration management system in the same package. A density goal of 100 latching MEMS switches/cm^2 was deemed a "modest" goal.
2) Development of a flexible wiring system architecture, based on prebuilt printed circuit boards (and eventually wiring harnesses) that would exploit an intelligently arranged distribution of terminals in blank (unpopulated) regions for mounting components, modules, boxes, etc. These boards would support the aforementioned multichip modules (MCMs) with the necessary daisychain connections and host interface to support external configuration.
3) The development of offline and eventually embeddable routing synthesis tools set to compute and transmit desired wiring configurations to the adaptive wiring manifold system. The tools, consistent with those for previous FPID devices, would accept as inputs the adaptive manifold detailed configuration and the user netlist specification, generating as output a bitstream sequence of switch closures.

For practical use, such adaptive wiring systems would ideally support other useful features.

1) Dynamic reprogrammability, to support in-system changes, especially for self-healing/fault recovery, as well as temporary diagnostic probes (which could be implemented either in development or even remotely in a fielded system). This feature would require the MEMS switches be reversible and reprogrammable many times. As the original notions were that this *in situ* reconfiguration would be done episodically (as opposed to continuously), it was felt arbitrarily that the ability to support several thousand switch cycles would be sufficient over a typical product life. This led to the definition of switches supporting "configuration grade" use (10^3 cycles) as opposed to continuous operation (ideally infinite cycles). The desire for *in situ* operation would bring other challenges, since arcing phenomenon effects could sharply limit the lifetime of typical relay switches, especially for devices with microscopic contact geometries.
2) The ability to diagnose the state of current configurations in a nonvolatile array. Since it was envisioned that smart wiring assemblies could be used in production systems and that the wiring patterns were set semipermanently, it would be undesirable to refresh the configuration of thousands of MEMS switches (which manage the connections of power distribution networks as well as other types of signals) upon reset. As such, a way of being able to non-intrusively read or capture state information would be useful.

Since often MEMS and ASICs are created with distinct semiconductor processes, the MEMS-based FPIDs would be implemented as multichip modules in a ball grid array (BGA) package, using an arrangement shown in Fig. 11, involving one or more MEMS switch crosspoint arrays (with an empirically chosen switch populations strategy) and a controlling sidecar ASIC to provide a host JTAG [47] control interface and to generate the detailed time sequenced voltage patterns necessary to isolate its set of individual switches. As several such FPID MCMS would be involved, the sidecar ASIC would be designed to support daisychain connections to permit configuration as a single extended bitstream. To capture student information (as well as any programming firmware for the sidecar ASIC), a small nonvolatile memory would be included in the MCM floor plan.

B. First Generation

A first generation in adaptive wiring research (~1994–2002) explored how MEMS switches could be integrated into wiring structures that themselves were integrated into the structural panels, in effect the first attempt (known

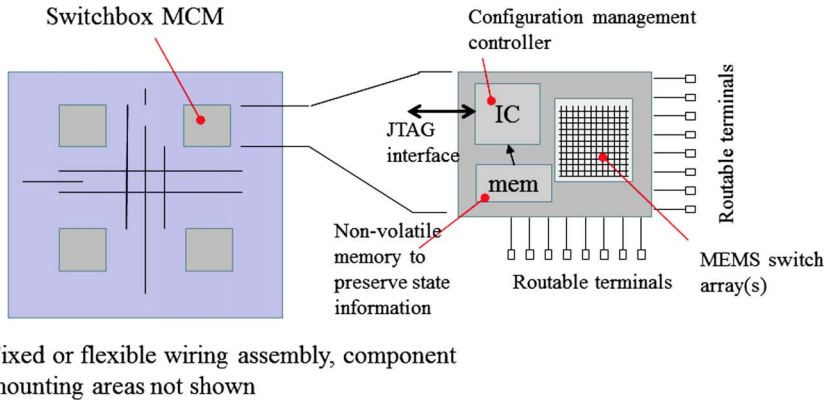


Fig. 11. ***Multichip module implementation of MEMS-based FPID. This module contains one or more MEMS switch crosspoint arrays and a controller sidecar ASIC.***

to the authors) to create a field-programmable wiring harness.

1) Switch Research: The switches in development for the first-generation panel project targeted the following properties:

- nonvolatile, having persistent (open and closed) bistable properties;
- low contact resistance;
- wide bandwidth, including dc operation;
- high current handling capacity (several hundred milliamperes);
- ability to support live switching;
- cycle life $>$ 10 000 operations.

During this period (mid-1990s), there were three dominant classes of MEMS approaches for actuation: electrostatic [23], piezoelectric [48], and thermal [24]. Electrostatic approaches usually involve high-voltage (but low-amperage) actuation, which complicate implementation in traditional CMOS circuitry. Electrostatic actuators, furthermore, are not optimized for high contact force, usually resulting in poor metal-to-metal contacts and therefore dc performance. Countering low contact resistance usually involves using high force, long throw actuation mechanisms. Piezoelectric actuation mechanisms, while having potential, were not explored. Thermally actuated MEMs switch configurations were deemed a good fit for this work, since they supported CMOS-level actuation voltages, high contact forces, and bistability. One promising configuration is shown in Fig. 12 that appears to address most of these requirements. This switch achieved bistability through two long-throw lateral thermal actuators, one to set the switch into each stable state, as shown in open [Fig. 13(a)] and closed [Fig. 13(b)] configurations. Unfortunately, the thermal MEMS relays explored in the early work, while promising, suffered from undercutting effects during manufacture that rendered the actuation mechanisms ineffective. While the contacts themselves

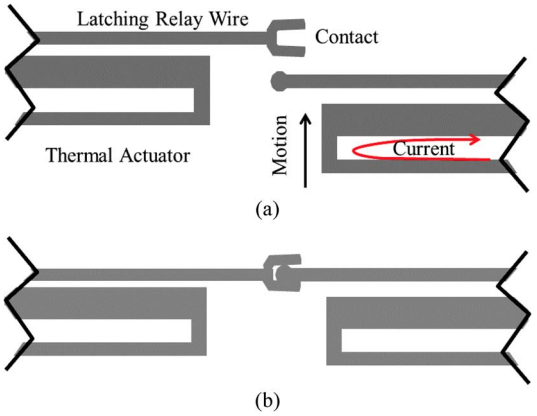


Fig. 12. ***Bistable MEMS switch design with two lateral thermal switches. (a) Open. (b) Closed.***

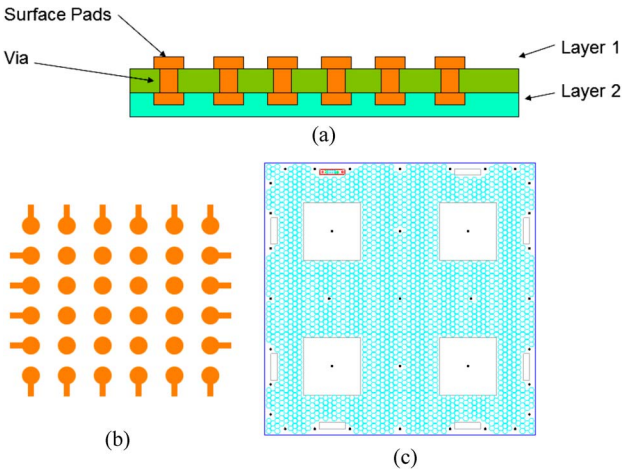


Fig. 13. ***The "liquid manifold" packaging concepts. (a) Component level land grid array (LGA). (b) Panel LGA pattern. (c) Panel structure design with mounting zones for modules.***

were viable (verified through manual actuation), the switch design was abandoned.

2) Architecture Research: Even in the earliest work, physical panel concepts were chosen to conform to a notion of multifunction structures [49], in which the structures necessary to build platforms (such as a spacecraft) would also support other functional roles. These panels could be thought of as a load-bearing circuit board panels, and it was aim in early research to make these smart wiring panels reconfigurable. A typical configuration research in this period involved some features shown in Fig. 13, including land grid array (LGA) mounting surfaces for modules [Fig. 13(a)] as well as the panel [pattern shown if Fig. 13(b)]. As suggested in our earlier discussion, mounting areas would be designated on the panel [Fig. 13(c)] for the attachment of modules. Some of the early architecture research to accompany the physical structures was referred to as "liquid manifold" [50], in which dynamic permutation routing switch boxes were demonstrated using Xilinx FPGAs, in effect embedding one programmable structure within another, the former being controlled by simple commands.

3) Demonstration Implementations: Due to the inability to obtain suitable MEMs switches, conventional macroscopic discrete latching relays were used to implement a simple multitiered wiring system design (supporting episodic reconfiguration as opposed to dynamic), prototyped on printed circuit boards. The control circuitry included features to keep track of switch state, so that it would be possible to query wiring system configuration details. This implementation, described in [47], while not remarkable in breaking practical ground, provided useful insights for subsequent work.

C. Second Generation

In a second generation (∼2002–2010) of adaptive wiring research, AFRL-sponsored research explored more advanced MEMS switches and an improved understanding of key architectural issues, such as wiring demand. Panel research evolved in parallel to form a construction methodology used in a line of conceptual research spacecraft.

1) Switch Research: Renewed attempts were made to create suitable MEMS devices. A primary concern were creating nonvolatile bistability mechanisms with high-quality contact surfaces, leading to dedicated research activities to study long throw actuators [51] and nanoscale contact surface structure [52]. To address contact degradation during live switching, purchased to mitigate arc formation were examined, such as using sacrificial contacts, either with an "arc gettering" whisker or entire dedicated redundant switches (to draw the arc, thus preserving contact surface degradation in the primary switch). A MEMS switch design, based on electromagnetic actuation, was developed [53] (through a concept proposed by Magfusion) as a promising candidate. While the switches did not provide new answers to live switching reliability, they did exhibit adequate current handling and bistability [54].

2) Architecture and Panel Research: The panel architecture research conducted during this time involved a set of eclectic studies, including investigations of wiring demand and routing architecture studies.

Approximately 600 nets comprising the TacSat2 spacecraft wiring harness database were measured and histogrammed, revealing a power law relationship analogous to Rent's rule relationship previous described for microelectronics. Even to the present, the authors are not aware of other analysis work done on wiring demand in spacecraft or other vehicular platforms, and it may therefore be premature to conclude that the scale-free model universally applies to is broader class of wiring system designs.

Work sponsored at Cambridgethrough a U.S. Air Force (USAF) research grant led to the discovery of a simple but elegant methodology for self-healing wiring systems that employ dynamically reconfigurable routing [55]. The concept involves performing a periodic sweep through a wiring system in which each switch is tested for continuity (connections involving live wires are bridged redundantly before separating them in the test). When bad switches are found, they are removed from the associated wiring resource graph.

Additional work began to explore embedding wiring systems into the structural panels of a spacecraft in support of research in plug-and-play spacecraft. This work was based on extending earlier research to develop pegboard-like panels for spacecraft. Wiring assemblies to accommodate the distribution of electrical power and data connections (to nonblocking Spacewire [56] packet routers) were embedded into each of several panels forming the boxlike structure of a spacecraft. It was envisioned that such panels can be inventoried for the on-demand construction of spacecraft. Components would be pulled from inventory and mounted to panels using the standard grid convention, aligning to and blind mating with connectors in an attempt to realize a cableless (or hidden cable) spacecraft.

3) Demonstration Implementations: A second-generation wiring panel system, referred to as adaptive wiring manifold 2 (AWM2), was created using ∼250 Magfusion bistable electromagnetic relays distributed over two large circuit boards. Connected together, these two AWM2 panels were demonstrated to form nonvolatile wiring connections for power and high-speed digital signals.

A parallel set of adaptive wiring designs were explored for the space plug-and-play architecture (SPA) [57] research program in which spacecraft were rendered as a set of structural "smart" panels (onto which modular components were mounted). A sequence of three demonstration systems were developed. The first of these, referred to as

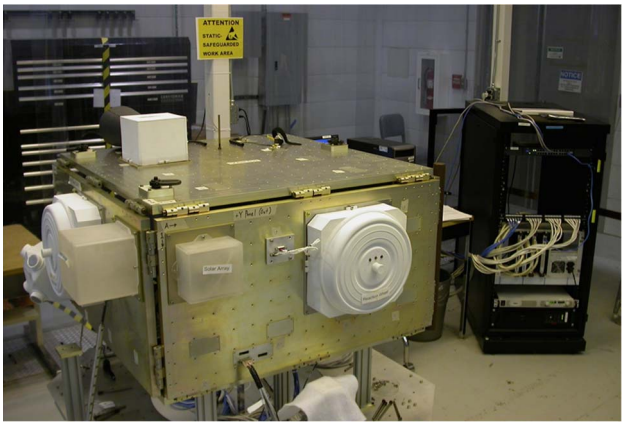


Fig. 14. *A spacecraft "concept bus," used to study modularization of plug-and-play spacecraft with integrated wiring, illustrating pegboard panel conventions, and a number of mounted components. Equipment rack in background operated test bypass circuitry.*

the concept bus, implemented a six-panel box demonstration (Fig. 14), using mock spacecraft modules (with electrically active interfaces to study plug-and-play network concepts). Routing and configuration management electronics were contained in separate boxes that were attached within the interior. The second demonstration system, referred to as plug-and-play satellite (PnPSat) [58] recessed routing circuitry inside the panels for power, data, and testing circuitry. A third demonstration (PnPSat 2) [59] did not introduce new concepts, but involved a hexagonal eight-panel (versus the six-panel PnPSat) configuration with electronics ruggedized for use in space environments. In each of the SPA panel designs, modules were surface mounted on a 5-cm x–y pegboard grid convention. As shown in Fig. 14, both blind-mated (modules cover and complete a socket connection when mechanically attached) and jumpered (module placed near a socket and connection is completed with a short cable) connection styles were supported. This particular blind mating strategy was ineffective, as it limited geometric flexibility in module placement and the self-aligning notion of the mechanical grid did not work well in practice.

Each panel design provided eight component docking ports, supporting power [60], data, and test connections. Each of the adaptive panel structures in the SPA research project comprised a limited, multidomain adaptive wiring design. As shown in Fig. 15, each panel featured a number of ports, most intended for connection by surface-mounted modules, with two sites reserved for interpanel connections. Each port contained connections to three limited domains, the first being power (only 28VDC), the second being a Spacewire [56] packet routing link, and the third being a serial testing interface, which allowed system-wide injection/query operations in a manner analogous to a JTAG [47] interface. In principle, systems could be composed of many panels, but geometries for simple spacecraft dictated a limited number (e.g., six or eight, typically).

D. Third Generation

Two trends marked the third generation of adaptive panel research (~2010–2014): 1) increased flexibility and pin mapping; and 2) movement to distributed, fine-grained (cellular) adaptive panel tiles. Some switch research was carried out, though no viable high-density MEMS switches were employed in any of the demonstrations.

1) Switch Research: Soon after the AWM2 demonstration, the switch vendor left the market. Several attempts were made to find alternate vendors, and some of these companies also stopped producing MEMS switches. Some work continued on alternate electrostatic switches, despite their aforementioned limitations. AFRL was also involved with DARPA on some promising new switch designs, based on liquid metal (e.g., gallium [61]) contacts. Such switches appear to solve the contact reliability problem [62], since the contact surface is continuously reformed. However none of these alternatives were available in sufficient quantities, and instead, we employed the use of tiny (~2 mm × 2 mm) solid state relays in the adaptive panel demonstrations.

2) Architecture and Panel Research

a) Flexible pin mapping: The ability to flexibly blind mate black box modules to panels (after the strategy suggested in Fig. 5), while not an essential requirement in field-programmable wiring systems, complements many of the same intrinsic goals: flexibility, rapid systems formation. It also reduces exterior cabling, though not necessarily a decrease in overall systems complexity (owing to the increased overhead necessary to programmably map pins between modules and panels). Despite the increase in

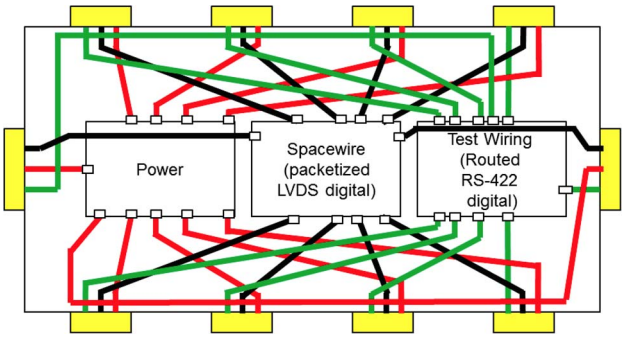


Fig. 15. *Simple, multidomain panel design used in space plug-and-play architecture (SPA) research. In this example, each panel contains a number of ports. Each port contained connections to three limited domains: 1) power (only 28VDC), 2) a Spacewire [56] packet routing link, and 3) a serial testing interface, which allowed system-wide injection/query operations in a manner analogous to a JTAG [47] interface.*

complexity, it could also improve reliability, since having programmable pin mappings allow for the inclusion of redundancy in both panels and modules, therefore providing non-unique opportunities for such mappings. In contrast, traditional pin and socket assemblies do not typically have the ability to cover from a failed connector. Adaptive pin mapping in field-programmable systems encompasses mechanical attachment (locations were modules may be physically mounted), electrical discovery, and pin mapping. The blind mating approach shown in Fig. 14 is the most brittle form, requiring mounting in a specific location and orientation (but providing robust attachment through traditional fasteners and electrical connectors). Overcoming these limitations requires the introduction of new concepts, and many are based on grid conventions, in which the periodic lattice of surface points provide opportunities for electrical and/or mechanical connection. Approaches such as "autoconnect" [63] suggested a Velco-like press-fit connection, for example. Other possibilities include the use of pin grid arrays (PGAs) or LGAs, the latter combined with interposers along with mechanical fasteners to provide force necessary to maintain good electrical contact. When serviceability is not a concern (i.e., modules once placed are permanently mounted), other options (such as ball grid array and underfilling) may be attractive.

In a study targeted for PnPSat [64], a design was developed to replace panels (such as Fig. 14) containing a distribution 24 mounting sites were allocated as a distributed contact array system. Each site employed a contact distribution as shown in Fig. 16, with a symmetry to accommodate any 90° rotation. While the work was promising, only limited work was completed toward this demonstration panel design in the second-generation activities.

b) Cellular tile implementation of adaptive wiring panels: For the core developments of the third-generation adaptive wiring panel research, we turned our attention to more generalized (beyond PnPSat) implementations of the cellular wiring strategies suggested in Fig. 17. We arrived at the following set of concepts [18], [65], [66]: 1) cell units (the basic tile building blocks); 2) cell management units (CMUs); 3) modules represented components that plug onto tiled arrays; and 4) some notion of a global system to orchestrate tile collections and mapping netlists to the tiles. These generalized schemes focused on solving free-form programmable wiring problems, in multiple domains (digital, power, analog, microwave), not the more limited ones exemplified in the PnPSat research. While a more detailed exposition of these concepts as they apply to specific embodiment is given in [18], [65], and [66], we provide an abstracted representation of the concepts here.

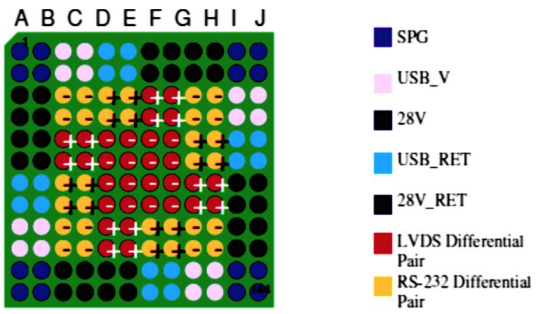


Fig. 16. ***Pin conventions for distributed contact arrays for prospective use in PnPSat [64].***

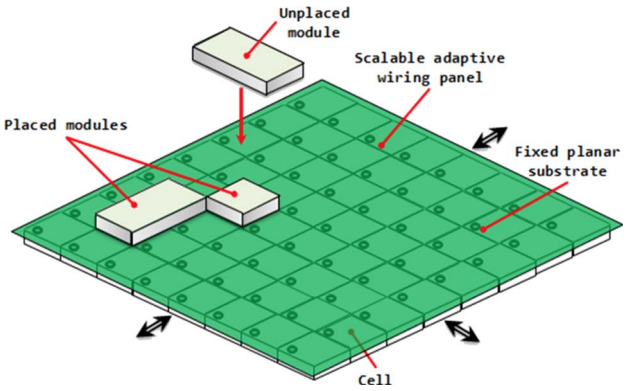


Fig. 17. ***Adaptive wiring panel (AWP) concept [18], [65], [66]: 1) cell units (the basic tile building blocks); 2) cell management units (CMUs); 3) modules represented components that plug onto tiled arrays; and 4) some notion of a global system to manage the set of tiles and to netlist the modules.***

The general cell architecture consists of the external physical interface and the functional interface. One approach to the physical interface is shown generically in Fig. 18, in terms of the tileable surface presented for connection by surface-mounted components. Here, we show the surface as being the intercalation of three grids: 1) a mechanical mounting grid (the coarsest grid); 2) a power terminal grid; and 3) a signal grid. The ratio and spacing of contacts in each grid can be set according to wiring supply and demand considerations. The functional unit in a generic sense is shown in Fig. 19. Each unit cell contains: 1) the CMU; 2) a pool of wiring resources, including switch matrices that may be differentiated by functional domains or other considerations; and 3) a set of terminals, with distinct terminals for surface-mounted connection and other terminals for intercell routing.

The CMU, which controls the switch configurations responsible for setting wiring pathways, manages three distinct types of communications. The first of these is a global interface, which manages commands for setting partial netlist assignments (i.e., a portion of a master overall netlist mapped over all titles), conveying the status of detected surface-mounted modules, and other aspects of health and status relating to unit cells. The second type of communication is intercell, managed in our work through dedicated links to each nearest-neighbor cell. The final

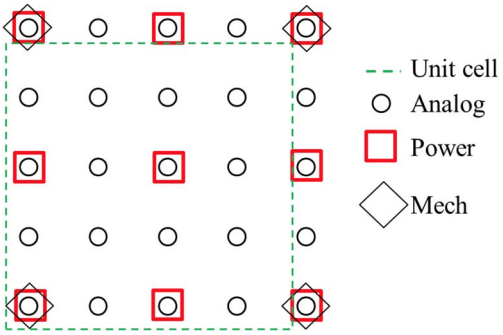


Fig. 18. ***Generic tile array conventions for mechanical, power, and signal grids.***

type of communications is a special accessory function, simply called a discovery interface. This interface establishes a systematic approach for detecting mounted modules and extracting their details.

We use the generic term module to denote surface-mounted components that conform to the conventions described here. A typical module will be designed to adhere to the mechanical, power, and signal distribution conventions of the generic panel system. It could be as simple as a single discrete component, such as a resistor, a battery, or a light bulb. It could also be a complex camera, a processor, or other electronic system. The modules may be small (fitting within a single tile) or may span multiple tiles. A number of mechanisms could be used to detect the presence, but we have adopted a discovery interface as being the most versatile method. This interface can be used to identify an arbitrary variety of details regarding surface-mounted components, but minimally would include physical aspect information (describing the surface shape reserved by the placed component) and identification and grid location of terminals. Other useful information could be embedded, including URLs to maintenance repositories, data sheets, etc.

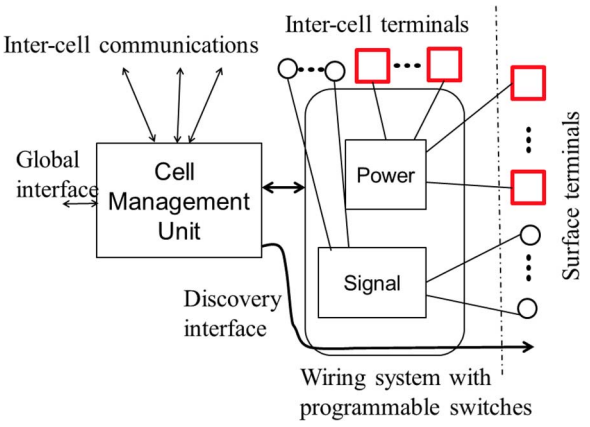


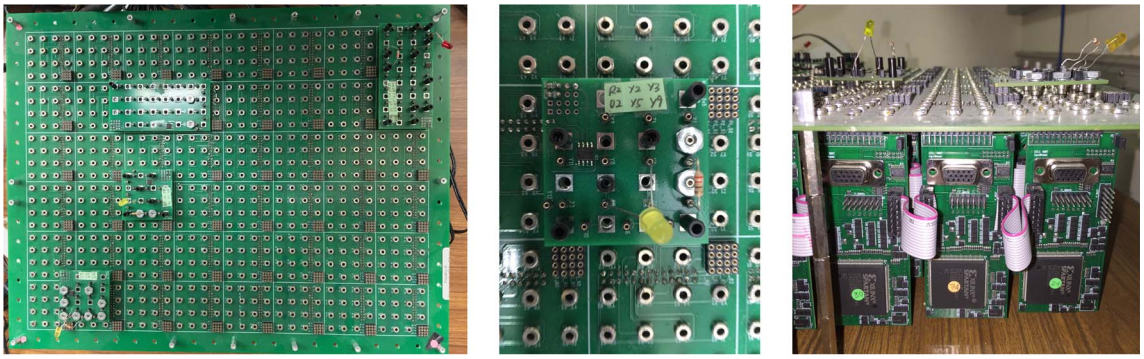
Fig. 19. ***Functional representation of unit cell. The unit cell transfer information with the CMU. The unit cell contains three grids: 1) a mechanical mounting grid (the coarsest grid); 2) a power terminal grid; and 3) a signal grid.***

The role of the global unit is to coordinate a notion of over-arching netlist, manage placement and routing operations, as well as capture knowledge of the tile distributions and characteristics and adjust the configuration details as they pertain to tiles. There are many considerations in the development of the global management infrastructure, including: 1) the allocation of routing details to individual tiles; 2) support for dynamic discovery of placed modules and route reconfiguration (as opposed to static placement and routing); 3) ability to accommodate dynamic topology changes in tiles; and 4) ability to accommodate redundant modules (potentially ignored and unconnected until potential need). Our inclusion of a discovery interface presupposes a type of dynamic interaction, though this is not strictly necessary in the cases where static compilation is involved. We have also presumed the notion of an external tool managing the overall configuration of the tile set. A far more intriguing (but presently unexplored) possibility is a distributed implementation of these functions, in which a single tile or group of tiles negotiates these functions without need of an external master.

3) Demonstration Implementation: We implemented a 48-tile version of a cellular adaptive wiring system (referred to as AWM3) embodying many of the principles described in [18], [65], and [66], shown in Fig. 20. The surface of this AWP, implemented as a large printed circuit board, was created in a 6 × 8 configuration, shown in Fig. 20(a), with four placed modules. Connectivity and physical attachment were accomplished through PGAs (pins on modules, sockets on the panel contacts). The modules [Fig. 20(b)] were also implemented as smaller circuit boards, which could be flexibly arranged in arbitrary positions and rotations on the larger surface panel. An external computer implemented the global controller (managing the netlist translating and implementation of switch closures in tiles necessary to implement the desired wiring patterns), and i2c interfaces were used for all of the communication interfaces. Approximately 5000 discrete solid-state relays were used, leading to very large unit cells whose bulk was projected below the mounting surface for modules [Fig. 20(c)].

V. SUMMARY OF FIELD-PROGRAMMABLE WIRING SYSTEMS

A summary of the panel developments described in this paper is shown in Table 3, spanning approximately a two-decade period. Essentially two types of adaptive panels were developed. The first of these, the AWM series, were intended to support free form wiring of arbitrary signal and power conductors. Their sophistication grew from a single



(a) (b) (c)

Fig. 20. *Images of the final AWP prototype [65]. (a) AWP with some modules placed on top. (b) A single module. (c) Side view showing the circuitry, the panel, and the modules on top.*

panel to a cellular implementation involving 48 panels. The second type of adaptive panel, the PnPSat series, was designed to manage a very limited range of signals (the distribution of fixed voltage, along with specialized LVDS and RS-422 signaling). In all of these designs, panels were intended to hide interior complexity, presenting a planar mounting surface to support mechanical and electrical attachment of modules. In early versions (up to AWM2), no discovery features were supported for modules. Although the panels supported dynamic reconfiguration, all switching actions were manually interpreted, so that any changes in component placement would not be automatically detected. Later adaptive panel work supported the automatic discovery placed modules, such that it is possible to relocate modules, place redundant copies, and support scaling to much larger systems.

The variety of switch types encountered in the AFRL-led research is summarized in Table 4. While the original motivation for the physical implementation of adaptive wiring was based on the promise of MEMS technology, only one adaptive wiring panel (AWM2) ever employed them. Even in this case, the use of the switch was problematic (e.g., sensitive alignment often failing after multiple solder reflow cycles). The original density goal was $> 100/\text{cm}^2$, but the switches used were $20\times$ larger and packaged form. Attempts to improve upon these limitations were unsuccessful, and the company left the marketplace. The pursuit of MEMS switches remained bifurcated from the architecture research. The subsequent developments have unfortunately shown little improvement, even nearly 20 years later. While MEMS technology has been in a broader sense very successful, little progress has effectively been made in creating practical MEMS switches with adequate density, bistability, and reliability. In AWM3, as a result, thousands of compact (2 mm $\times$ 2 mm) solid-state relays were employed. While this led to a successful demonstration system, the overhead in the most optimistic sense (in terms of bulk and power consumption) is unsuitable for practical use.

VI. RECOMMENDATIONS FOR FUTURE WORK

There are significant challenges associated with the development of the technologies that will enable the next generation of field-programmable wiring systems. We provide a brief discussion of recommendations for future work in this area.

Table 3 Comparison of Programmable Wiring Systems

Year	Project	Domains	Switch	Module Mount	Panels	Switches
1994	Aptix	Digital (<100 MHz) and small-signal analog	FPID	PGA	1	many
1999	Liquid Manifold	Digital	FPGA	LGA	1	many
2000	AWM1	Digital, analog, and power	Latching relay	Socket	1	dozens
2005	AWM2	Digital, analog, and power, radio-frequency	Latching MEMS relays	Socket	2	~250
2006	Concept Bus	LVDS, RS-422, and 28VDC power	Spacewire, test data and power routing	Socket	6	6
2008	PnPSat 1	LVDS, RS-422, and 28VDC power	Spacewire, test data and power routing	Socket	6	6
2010	PnPSat 2	LVDS, RS-422, and 28VDC power	Spacewire, test data and power routing	Socket	8	8
2013	AWM3	Digital, analog, and power	Solid state relays	PGA	48	~5000

Table 4 Switch Types Examined

Type of Switch	Supplier	Signal Types	Class	Density	Comments
FPID	Aptix	Digital, small-signal analog	Volatile	Highest	Used in prototyping system.
FPGA	Various	Digital	Volatile	Highest	High density, limited to digital.
Conventional Relay	Various	Broad range	Latching	Lowest	Used in AWM1. Low density.
MEMS thermal	Experimental	Broad range	Latching	Medium (>100 cm^2)	Actuator failed, contact system worked.
MEMS magnetic	MagFusion	Broad range	Latching	Low (5mm×5mm	Used in AWM2. Vendor left market.
MEMS liquid metal	Honeywell (experimental)	Broad range	Volatile	Low-Medium	Superior for live switching, DARPA funded, switches not brought to market.
Solid state relay	Panasonic	Broad range	Volatile	Low (2mm×2mm)	Used in AWM3.
MEMS electrostatic	Various (experimental)	Varies, some have limited DC performance	Volatile	Low-Medium	Complicated support, limited availability.

A. Switching Technologies

We have the following list of challenges for switching technologies.

- Low-cost, low-energy relays: The key concept here is to reduce the cost and energy requirements of the relays so as to enable the development of larger designs that require large numbers of relays. Overall, the goal here will be to provide scalable methods for routing power and analog signals that can match the success of routing digital signals in current FPGA devices.
- Effective multipurpose switches: Currently, we have the requirement to use different types of switches that can route different signals. As a result, field-programmable wiring systems can be limited by the spatial arrangement of the different types of switches. Clearly then, the development of multipurpose switches can greatly simplify the field-programmable wiring systems design by eliminating the need to predetermine the distribution of different switch types prior to deployment.
- Switches for time-shared interconnections: A single switch can be designed to support multiple interconnections. For example, to maintain a potential level using a power connection, we could time-share to recharge capacitor circuits. As long as the voltage level can be maintained within certain tolerances, a single power input would then be able to route power signals to several power terminals.

Following the development of effective switching technologies, we will have development of efficient packaging configurations. For implementations of multidomain fabrics, there is a need to investigate the distribution and topology of switching types that can support different types of signals. Techniques from survivable network design can provide a framework that supports recovery from connection failures [8], [67]. Effective digital signal topologies such as the methods described by DeHon [33] will need to be extended to support multidomain fabrics.

B. Component Failure Detection and Replacement

As with our description of the AWP, the goal has shifted from simply interconnecting components to one of maintaining the integrity of circuits. To support the proper circuit behavior, we need to have full support for replacing faulty components. A basic plan for replacing faulty components involves the following steps.

- Equivalent component library sensing: The current implementation of the AWP [65] supports automated module component sensing. To support component replacement, the goal will be to determine the location of equivalent components that can be used to replace each other.
- Working component failure detection: In order to detect component failure, we clearly have to measure component characteristics (e.g., voltage and current) during operation. Then, a significant deviation from normal operating values can be used as a detector of component failure.
- Failed component replacement through rerouting: Once component failure has been detected, an equivalent component can be used to replace it.

C. Circuit Integrity Management

To maintain circuit integrity in future panels, there is a clear need to implement the following functionalities.

- Circuit connection diagnostics: The goal here is to detect connection failure.
- Circuit shutdown: Once a faulty connection has been detected, the safest step will be to shutdown the circuit to prevent damage.
- Circuit reboot: Each circuit will then have to implement a reboot mechanism to support recovery.
- Live circuit recovery using dynamic partial reconfiguration: This is an extension of the current AWP system [65] that supports recovery from module replacement. We also borrow the term "dynamic partial reconfiguration" from FPGAs to emphasize live circuit recovery without the need to shutdown

and reboot. Thus, if possible, live circuit recovery would be preferred from the requirement to shutdown and reboot.

- Circuit connection rerouting: Once a failed connection has been detected, the connection will need to be rerouted (if possible).
- Circuit component replacement: Based on the component diagnostics, we need to provide the ability to replace faulty components using equivalent components.

D. Circuit Operation Management

The current implementation of the AWP cannot recover from failure of the CMUs. In future research, it is important to consider the development of technologies that can support recovery from CMU failures.

- Remote circuit management: Starting from the fact that the most basic connectivity problems are NP-hard, it is clear that implementing larger circuits become prohibitively expensive. To support future growth, it makes sense to avoid the limitations associated with the computational resources of the programmable wiring panels. Instead, we will need methods to communicate the problems to larger scale computational systems that can then communicate the solutions. We use the term "remote circuit management" to refer to this framework.
- CMU operational failure detection: The failure of the CMU hardware needs to be immediately detected. However, it is important to note that we also need to detect software failures. When using remote circuit management, we can employ larger computational resources to attempt recovery from software failures.
- Parallel and distributed circuit management: The support of multiple CMUs will allow recovery from single CMU failure. Here, a simple polling protocol can be used to detect failures. Furthermore, the presence of multiple CMUs can support the development of parallel and distributed algorithms. We note that we can start from existing parallel routing algorithms based on the shortest path (e.g., see [68]). To support recovery from failure, future research can adopt ideas from the MapReduce framework [69].

Ultimately, these concepts will need to be adopted to the topologies that will support multidomain fabrics. Even more interestingly, it will be interesting to investigate topologies that can support effective algorithms.

VII. CONCLUSION

The paper provided an extensive review of the fundamental concepts and principles associated with the development of field-programmable wiring systems. The development of such systems has greatly benefited from seemingly independent, yet interrelated technologies. A list of important, enabling technologies includes: 1) the development of switches that support different signal types; 2) network topologies that can support complex interconnections; 3) practical routing algorithms; and 4) the recent integration of many of these basic technologies using smart module sensing and rerouting in the development of three generations of field-programmable wiring systems. We expect that the future development of field-programmable wiring systems will benefit from advancement in these enabling technology areas.

In terms of hardware, the most urgent need is the development of effective, low-cost, and low-energy consumption relays that can be used for routing power and analog signals. In terms of software, we expect the recent developments in parallel and distributed algorithms to greatly benefit the future generations of CMUs. ■

REFERENCES

[1] S. Hauck, "The roles of FPGAs in reprogrammable systems," *Proc. IEEE*, vol. 86, no. 4, pp. 615–638, Apr. 1998.

[2] R. J. Francis, J. Rose, and Z. G. Vranesic, *Field-Programmable Gate Arrays*, vol. 180. New York, NY, USA: Springer-Verlag, 1992.

[3] I-Cube, "Iqx family data sheet," Jan. 1999. [Online]. Available: http://www.ic72.com/pdf_file/i/20685.pdf

[4] R. F. Hartmann, S. J. Kopec, Jr., H.-C. So, and S.-C. Wong, "Programmable logic device with array blocks connected via programmable interconnect," U.S. Patent 4 871 930, Oct. 3, 1989.

[5] M. J. Wale and C. Edge, "Self-aligned flip-chip assembly of protonic devices with electrical and optical connections," *IEEE Trans. Compon. Hybrids Manuf. Technol.*, vol. 13, no. 4, pp. 780–786, Dec. 1990.

[6] J. C. McDonald, S. J. Metallo, and G. M. Whitesides, "Fabrication of a configurable, single-use microfluidic device," *Anal. Chem.*, vol. 73, no. 23, pp. 5645–5650, 2001.

[7] C. Leiserson and F. Maley, "Algorithms for routing and testing routability of planar VLSI layouts," in *Proc. 17th Annu. ACM Symp. Theory Comput.*, Dec. 1985, pp. 69–78.

[8] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Mineola, NY, USA: Courier Dover, 1998.

[9] Y. Wu, S. Tsukiyama, and M. Marek-Sadowska, "Graph based analysis of 2-D FPGA routing," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 15, no. 1, pp. 33–44, Jan. 1996.

[10] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[11] T. H. Cormen *et al.*, *Introduction to Algorithms*, vol. 2. Cambridge, MA, USA: MIT Press, 2001.

[12] J. Roy and I. Markov, "Seeing the forest and the trees: Steiner wirelength optimization in placement," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 4, pp. 632–644, Apr. 2007.

[13] F. K. Hwang, D. S. Richards, and P. Winter, *The Steiner Tree Problem*. Amsterdam, The Netherlands: Elsevier, 1992.

[14] M. R. Garey and D. S. Johnson, "The rectilinear Steiner tree problem is NP-complete," *SIAM J. Appl. Math.*, vol. 32, no. 4, pp. 826–834, 1977.

[15] E. Ihler, G. Reich, and P. Widmayer, "Class Steiner trees and VLSI-design," *Discrete Appl. Math.*, vol. 90, no. 13, pp. 173–194, 1999.

[16] M. J. Alexander and G. Robins, "New performance-driven FPGA routing algorithms," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 15, no. 12, pp. 1505–1517, Dec. 1996.

[17] S. Lei and W. K. Mak, "Simultaneous constrained pin assignment and escape routing for FPGA-PCB codesign," in *Proc.*

IEEE Int. Conf. Field Programm. Logic Appl., Sep. 2011, pp. 435–440.

[18] V. Murray *et al.*, "Cell-based architecture for adaptive wiring panels: A first prototype," *J. Aerosp. Inf. Syst.*, vol. 10, no. 4, pp. 187–208, 2013.

[19] H. Hosaka, H. Kuwano, and K. Yanagiswa, "Electromagnetic microrelays: Concepts and fundamental characteristics," in *Proc. Micro Electro Mech. Syst.*, 1993, pp. 12–17.

[20] D. J. Bishop, S. Jin, J. Kim, and A. G. Ramirez, "Non-volatile MEMS micro-relays using magnetic actuators," U.S. Patent 6 124 650, Sep. 26, 2000.

[21] K. Suzuki, "Micro electro mechanical systems (MEMS) micro-switches for use in dc, RF, and optical applications," *Jpn. J. Appl. Phys.*, vol. 41, no. 6S, p. 4335, 2002.

[22] G. M. Rebeiz, *RF MEMS: Theory, Design, and Technology*. New York, NY, USA: Wiley, 2004.

[23] J.-E. Wong, J. H. Lang, and M. A. Schmidt, "An electrostatically-actuated MEMS switch for power applications," in *Proc. IEEE 13th Annu. Int. Conf. Micro Electro Mech. Syst.*, 2000, pp. 633–638.

[24] M. Daneshmand, S. Fouladi, R. R. Mansour, M. Lisi, and T. Stajcer, "Thermally actuated latching rf MEMS switch and its characteristics," *IEEE Trans. Microw. Theory Tech.*, vol. 57, no. 12, pp. 3229–3238, Dec. 2009.

[25] A. M. Mohsen, "Interconnect substrate with circuits for field-programmability and testing of multichip modules and hybrid circuits," U.S. Patent 5 371 390, Dec. 6, 1994.

[26] D. Llamocca and M. Pattichis, "A dynamically reconfigurable pixel processor system based on power/energy-performance-accuracy optimization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 3, pp. 488–502, Mar. 2013.

[27] J. Li and C.-K. Cheng, "Routability improvement using dynamic interconnect architecture," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 6, no. 3, pp. 498–501, Sep. 1998.

[28] P.-H. Yuh, C.-L. Yang, and Y.-W. Chang, "Bioroute: A network-flow-based routing algorithm for the synthesis of digital microfluidic biochips," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 27, no. 11, pp. 1928–1941, Nov. 2008.

[29] V. Pasham, W. Moreno, and F. Falquez, "Field programmable multi chip modules using programmable laser interconnects," in *Proc. IEEE Int. Workshop Rapid Syst. Prototyping*, Jul. 1999, pp. 210–213.

[30] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 2, pp. 203–215, Feb. 2007.

[31] J. Cong and K. Minkovich, "Optimality study of logic synthesis for LUT-based FPGAs," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 26, no. 2, pp. 230–239, Feb. 2007.

[32] R. Wood and R. Rutenbar, "FPGA routing and routability estimation via boolean satisfiability," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 6, no. 2, pp. 222–231, Jun. 1998.

[33] A. DeHon, "Unifying mesh-and tree-based programmable interconnect," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 10, pp. 1051–1065, Oct. 2004.

[34] N. Weaver, J. Hauser, and J. Wawrzynek, "The SFRA: A corner-turn FPGA architecture," in *Proc. 2004 ACM/SIGDA 12th Int. Symp. Field Programm. Gate Arrays*, Monterey, CA, USA, Feb. 2001, pp. 3–12.

[35] A. Aggarwal and D. Lewis, "Routing architectures for hierarchical field programmable gate arrays," in *Proc. IEEE Int. Conf. Comput. Design, VLSI Comput. Processors*, Oct. 1994, pp. 475–478.

[36] K. Fujiyoschi, Y. Kajitani, and H. Niitsu, "Design of minimum and uniform bipartites for optimum connection blocks of FPGA," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 16, no. 11, pp. 1377–1383, Nov. 1997.

[37] Y.-L. Wu, D. Chang, M. Marek-Sadowska, and S. Tsukiyama, "Not necessarily more switches more routability [sic.]," in *Proc. Asia South Pacific Design Autom. Conf.*, Jan. 1997, pp. 579–584.

[38] E. Hung, J. B. Goeders, and S. J. Wilton, "Faster FPGA debug: Efficiently coupling trace instruments with user circuitry," in *Reconfigurable Computing: Architectures, Tools, and Applications*. New York, NY, USA: Springer-Verlag, 2014, pp. 73–84.

[39] J. Lyke and L. Owczarzak, Air Force Research Laboratory, personal communications.

[40] G. Parthasarathy, M. Marek-Sadowska, A. Mukherjee, and A. Singh, "Interconnect complexity-aware FPGA placement using rent's rule," in *Proc. Int. Workshop Syst.-Level Interconnect Prediction*, 2001, pp. 115–121.

[41] S. Takamaeda-Yamazaki and K. Kise, "Flipsyrup: Cycle-accurate hardware simulation framework on abstract FPGA platforms," in *Proc. 24th Int. Conf. Field Programm. Logic Appl.*, Sep. 2014, pp. 1–4.

[42] C. Chang *et al.*, "Rapid design and analysis of communication systems using the bee hardware emulation environment," in *Proc. 14th IEEE Int. Workshop Rapid Syst. Prototyping*, Jun. 2003, pp. 148–154.

[43] R. Arce-Nazario, M. Jimenez, and D. Rodriguez, "An assessment of high-level partitioning techniques for implementing discrete signal transforms on distributed hardware architectures," in *Proc. 48th Midwest Symp. Circuits Syst.*, Aug. 2005, vol. 2, pp. 1438–1441.

[44] S. Hauck, G. Borriello, and C. Ebeling, "Mesh routing topologies for multi-FPGA systems," in *Proc. IEEE Int. Conf. Comput. Design, VLSI Comput. Processors*, Oct. 1994, pp. 170–177.

[45] A. Ejnioui and N. Ranganathan, "Multiterminal net routing for partial crossbar-based multi-FPGA systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 1, pp. 71–78, Feb. 2003.

[46] S. Brown, J. Rose, and Z. Vranesic, "A detailed router for field-programmable gate arrays," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 11, no. 5, pp. 620–628, May 1992.

[47] J. Lyke, W. Wilson, and P. Contino, "Mems-based reconfigurable manifold," in *Proc. MAPLD Int. Conf.*, 2001. [Online]. Available: http://www.slideserve.com/theta/presentation-to-mapld-2001-11-13-september-2001.

[48] J. Y. Park, Y. J. Yee, H. J. Nam, and J. U. Bu, "Micromachined RF MEMS tunable capacitors using piezoelectric actuators," in *IEEE MTT-S Int. Microwave Symp. Dig.*, 2001, vol. 3, pp. 2111–2114.

[49] E. Fosness, J. Guerrero, K. Qassim, and S. Denoyer, "Recent advances in multi-functional structures," in *Proc. IEEE Aerosp. Conf.*, 2000, vol. 4, pp. 23–28.

[50] R. Abbott, "Adaptive computer systems," in *Proc. IEEE Aerosp. Conf.*, 2002, vol. 4, pp. 4-1819–4-1824.

[51] H. Jiao, "High force actuator for micro-spacecraft systems," Los Gatos Research, Mountain View, CA, USA, Tech. Rep. AFRL-VS-TR-2003-1120, Jun. 2003.

[52] J. Tringe, T. Uhlman, A. Oliver, and J. Houston, "A single asperity study of au/au electrical contacts," *J. Appl. Phys.*, vol. 93, no. 8, pp. 4661–4669, Apr. 2003.

[53] J. Shen, "Novel field programmable technology based on latching micro-electromagnetic switches," MagFusion, Inc., Chandler, AZ, USA, Tech. Rep. AFRL-VS-TR-2003-1181, Nov. 2003.

[54] "Magfusion's MEMS-based relay tackles space, cost issue," *EE Times*, Jun. 2003.

[55] S. Thompson and A. Mycroft, "Self-healing reconfigurable manifolds," in *Proc. Designing Correct Circuits*, 2006. [Online]. Available: http://www.cse.chalmers.se/~ms/DCC06/abstracts/AM.pdf

[56] European Cooperation for Space Standardization (ECSS) Secretariat, "Space engineering: Spacewire—Links, nodes, routers and networks," Jan. 24, 2003. [Online]. Available: http://snebulos.mit.edu/projects/reference/NASA-Generic/ECSS-E-40-12A.pdf

[57] J. Lyke, "Plug-and-play satellites," *IEEE Spectrum*, vol. 49, no. 8, pp. 36–42, Aug. 2012.

[58] D. Fronterhouse, J. Lyke, and S. Achramowicz, "Plug-and-play satellite (PnPSat)," in *Proc. AIAA Infotech@ Aerosp.*, 2007. [Online]. Available: http://arc.aiaa.org/doi/pdf/10.2514/6.2007-2914

[59] D. Fronterhouse, K. Center, B. Strunce, T. Mann, and J. Dipalma, "PnPSat-2 SPA technology testbed initial results and development status," in *Proc. IEEE Aerosp. Conf.*, Mar. 2010, pp. 1–12.

[60] W. Boncyk, "Developing a distributed power and grounding architecture for PnPSat," in *Proc. IEEE Aerosp. Conf.*, Mar. 2008, pp. 1–9.

[61] M. G. Wong, "Bending piezoelectrically actuated liquid metal switch," U.S. Patent 6 515 404, Feb. 4, 2003.

[62] P. Sen and C.-J. Kim, "A liquid-metal RF MEMS switch with dc-to-40 GHz performance," in *Proc. IEEE 22nd Int. Conf. Micro Electro Mech. Syst.*, Jan. 2009, pp. 904–907.

[63] P. Joshi, B. Decker, J. Magill, and M. Hinds, "Intelligent, universal, reconfigurable electromechanical interface for modular systems assembly," U.S. Patent 7 763 995, Jul. 27, 2010.

[64] E. van Doorn and W. Nicholas, "Hybrid wiring circuitry," Intelligent Automation, Inc., Rockville, MD, USA, Tech. Rep. AFRL-RV-PS-TR-2009-1169, Apr. 2010.

[65] D. Llamocca *et al.*, "Scalable open-source architecture for real-time monitoring of adaptive wiring panels," *J. Aerosp. Inf. Syst.*, vol. 11, no. 6, pp. 344–358, 2014.

[66] V. Murray, G. Feucht, J. Lyke, M. Pattichis, and J. Plusquellic, "Cell-based architecture for reconfigurable wiring manifolds," in *Proc. AIAA Infotech@ Aerosp. Conf.*, 2010. [Online]. Available: http://arc.aiaa.org/doi/pdf/10.2514/6.2010-3497

[67] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "Sndlib 1.0—Survivable network design library," *Networks*, vol. 55, no. 3, pp. 276–286, 2010.

[68] I. Foster, *Designing and Building Parallel Programs*. Reading, MA, USA: Addison-Wesley, 1995.

[69] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

ABOUT THE AUTHORS

Victor Murray (Senior Member, IEEE) received the B.S. (high honors, ranked first in the class) degree in electrical engineering from the Pontificia Universidad Católica del Perú, Lima, Peru, in 2003 and the M.S. and Ph.D. degrees in electrical engineering from the University of New Mexico (UNM), Albuquerque, NM, USA, in 2005 and 2008, respectively.

He is currently a Professor and Chair of the Department of Electrical Engineering at the Universidad de Ingeniería y Tecnología (UTEC), Lima, Peru, and Research Assistant Professor in the Department of Electrical and Computer Engineering (ECE), UNM. He has considerable experience in developing adaptive reconfigurable hardware, based on field-programmable gate arrays (FPGAs), for the Air Force Research Laboratory, and developing methods and algorithms in medical imaging for detecting diseases in collaboration to, mainly, VisionQuest Biomedical, Albuquerque, NM, USA and the University of Cyprus, Nicosia, Cyprus. His research interests include amplitude modulation-frequency modulation (AM-FM) methods, digital image and video processing, medical imaging, and hardware design using very high descriptive language (VHDL), and FPGAs.

Dr. Murray was the Research Director of the ECE's Image and Video Processing and Communication Lab (2009–2012). He was a recipient of the STC.UNM Innovation Award at UNM in 2014 and 2015.

Marios Pattichis (Senior Member, IEEE) received the B.Sc. (high honors and special honors) degree in computer sciences and the B.A. (high honors) degree in mathematics, both in 1991, the M.S. degree in electrical engineering in 1993, and the Ph.D. degree in computer engineering in 1998, all from the University of Texas at Austin, Austin, TX, USA.

He is currently a Professor with the Department of Electrical and Computer Engineering, University of New Mexico (UNM), Albuquerque, NM, USA. He was a founding Co-PI of COSMIAC at UNM. At UNM, he is currently the director of the image and video Processing and Communications Lab (ivPCL, ivpcl.org). His current research interests include digital image, video processing, communications, dynamically reconfigurable computer architectures, and biomedical and space image-processing applications.

Dr. Pattichis has been an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING and the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, and has also served as a Guest Associate Editor for the IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE. He was the General Chair of the 2008 IEEE Southwest Symposium on Image Analysis and Interpretation. He was a recipient of the 2004 Electrical and Computer Engineering Distinguished Teaching Award at UNM. For his development of the digital logic design labs at UNM he was recognized by the Xilinx Corporation in 2003 and by the UNM School of Engineering's Harrison faculty excellent award in 2006.

Daniel Llamocca (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the Pontifical Catholic University of Peru, Lima, Peru, in 2002 and the M.Sc. degree in electrical engineering and the Ph.D. degree in computer engineering from the University of New Mexico, Albuquerque, NM, USA, in 2008 and 2012, respectively.

He is currently an Assistant Professor at Oakland University, Rochester, MI, USA. His research deals with runtime automatic adaptation of hardware resources to time-varying constraints with the purpose of delivering the best hardware solution at any time. His current research interests include: reconfigurable computer architectures for signal, image, and video processing; high-performance architectures for computer arithmetic, communication, and embedded interfaces; embedded system design; and runtime partial reconfiguration techniques on FPGAs.

James Lyke (Senior Member, IEEE) received the B.S. degree in electrical engineering from the University of Tennessee, Knoxville, TN, USA, in 1984, the M.S. degree in electrical engineering from the Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, USA, in 1989, and the Ph.D. degree in electrical engineering from the University of New Mexico, Albuquerque, NM, USA, in 2004.

He was in active duty military service with the U.S. Air Force from 1984 through 1995. Since 1990, he has supported the Air Force Research Laboratory (AFRL), Space Vehicles Directorate (AFRL/RV), Kirtland Air Force Base, NM, USA, including its precursor organizations (Weapons Laboratory, 1990–1991, and Phillips Laboratory, 1991–1998), in a number of capacities. He is currently technical advisor to the AFRL Space Electronics Branch (Space Vehicles Directorate) and an AFRL Fellow since 2008. He has lead over 100 in-house and contract research efforts involving 2-D and 3-D advanced packaging, radiation-hardened microelectronics, and scalable, reconfigurable computational and systems architectures, with recent emphasis on modularity and the rapid formation of complex systems. He has authored over 100 publications (journal and conference papers, book chapters, and technical reports), four receiving best paper awards, and he has been awarded 11 U.S. patents.

Dr. Lyke is an Associate Fellow of the American Institute of Aeronautics and Astronautics (AIAA) and serves on the AIAA Computer Systems Technical Committee. He was selected as recipient of the Federal Laboratory Consortium award for Excellence in Technology Transfer in 1992 and for the U.S. Air Force Science and Engineering Award in Exploratory and Advanced Technology Development in 1997 and 2000.