Principles of Evolvable and Adaptive Hardware

IEEE 2007 Symposium Series on Computational Intelligence

Honolulu, Hawaii

April 1, 2007

John C. Gallagher
Department of Computer Science and Engineering
Department of Electrical Engineering
john.gallagher@wright.edu

IEEE Computational Intelligence Society

Introduction and Purpose

This tutorial is aimed at scientists, engineers, and technical managers interested in coming up to speed on both the practice and promise of Evolvable and Adaptive Hardware (EAH) methodologies. It will presume no prior EAH experience and will cover basic concepts including evolutionary algorithms and related optimization methods, basic digital and analog reconfigurable hardware systems and illustrative examples drawn from the literature. This tutorial will equip newcomers with the necessary background to fully participate in technical and applications oriented discussions to be held later in the WEAH workshop at this symposium.

Disclaimers and Warnings

- This tutorial is meant to be *representative,* not *exhaustive*. It will not cover the full breadth of the field. However, references and URLs will be provided so the interested person can track down all related works.
- This tutorial is meant for those with little or no experience with evolvable and adaptive hardware. Therefore it will touch on many issues that experts will find old hat -- my apologies to experts present in the room.

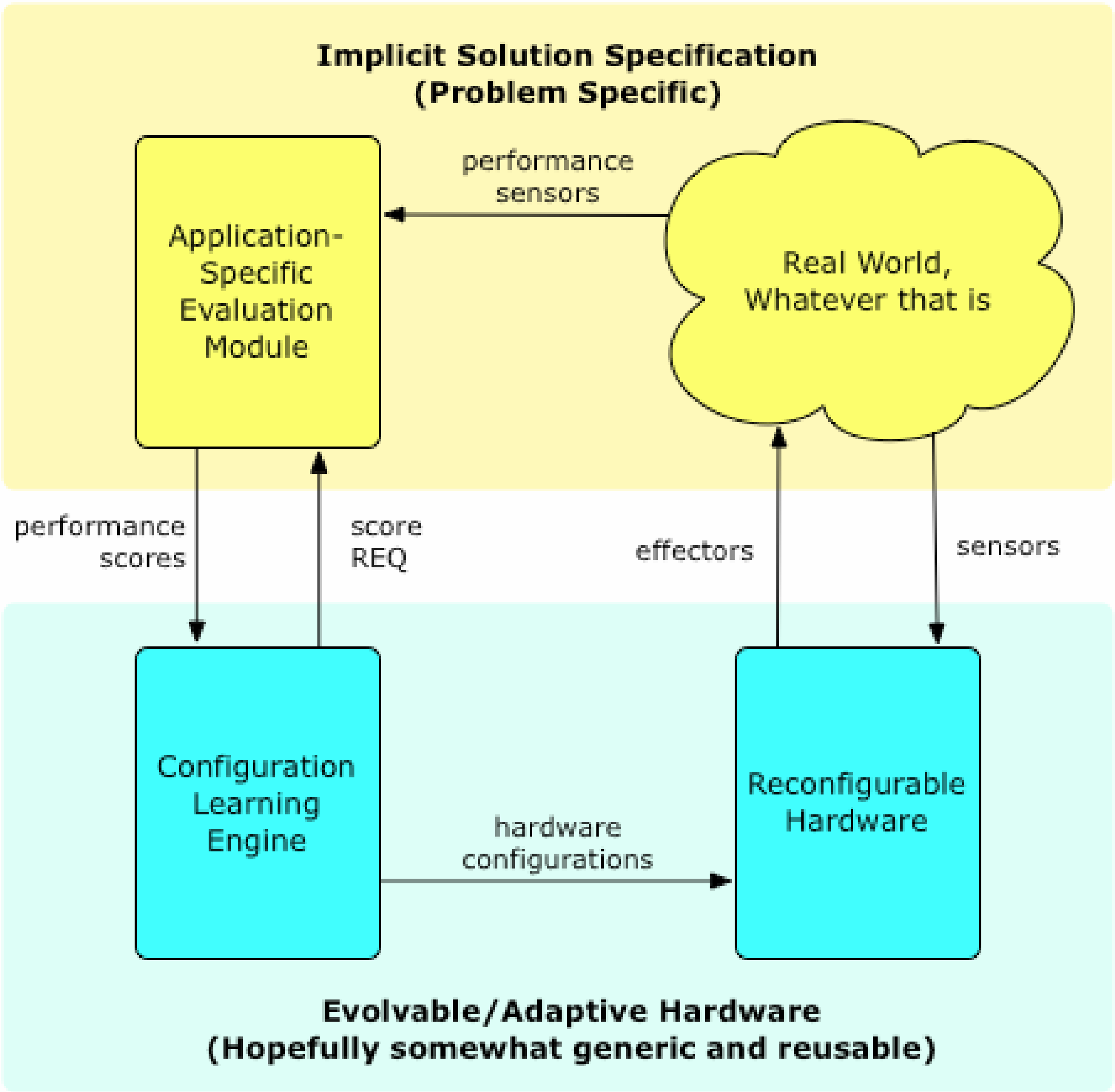
General Outline

- Basic Definitions
- Motivation - Why do this at all?
- Basic Techniques
- Big Open Questions

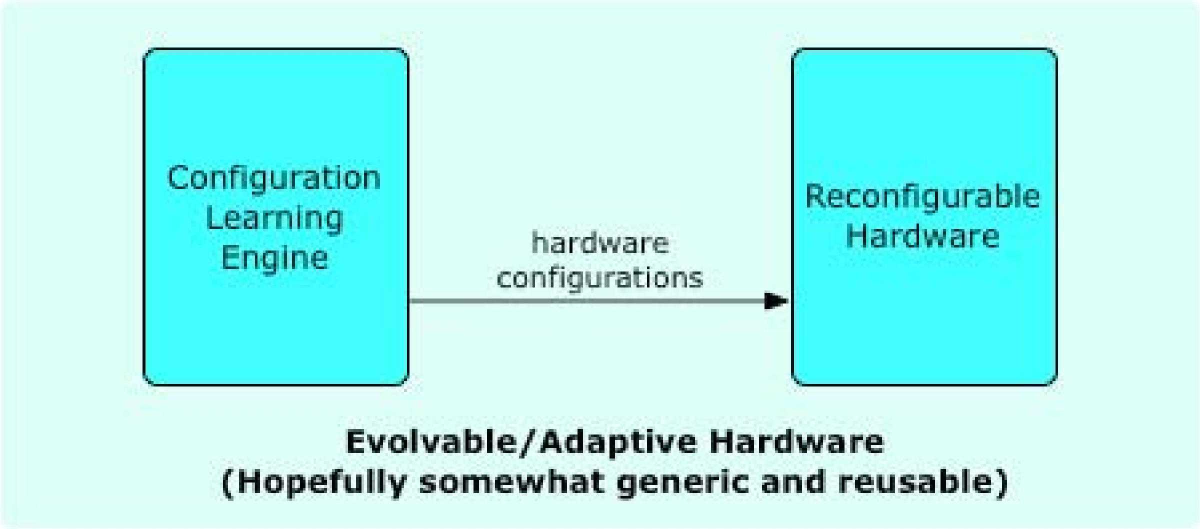
Basic Definitions

- *Evolvable and Adaptive Hardware* is reconfigurable hardware + configuration engine
- The *configuration engine* learns how to configure the hardware to accomplish some task based on observations of the performance of candidate configurations.

Evolvable and Adaptive Hardware (EAH)

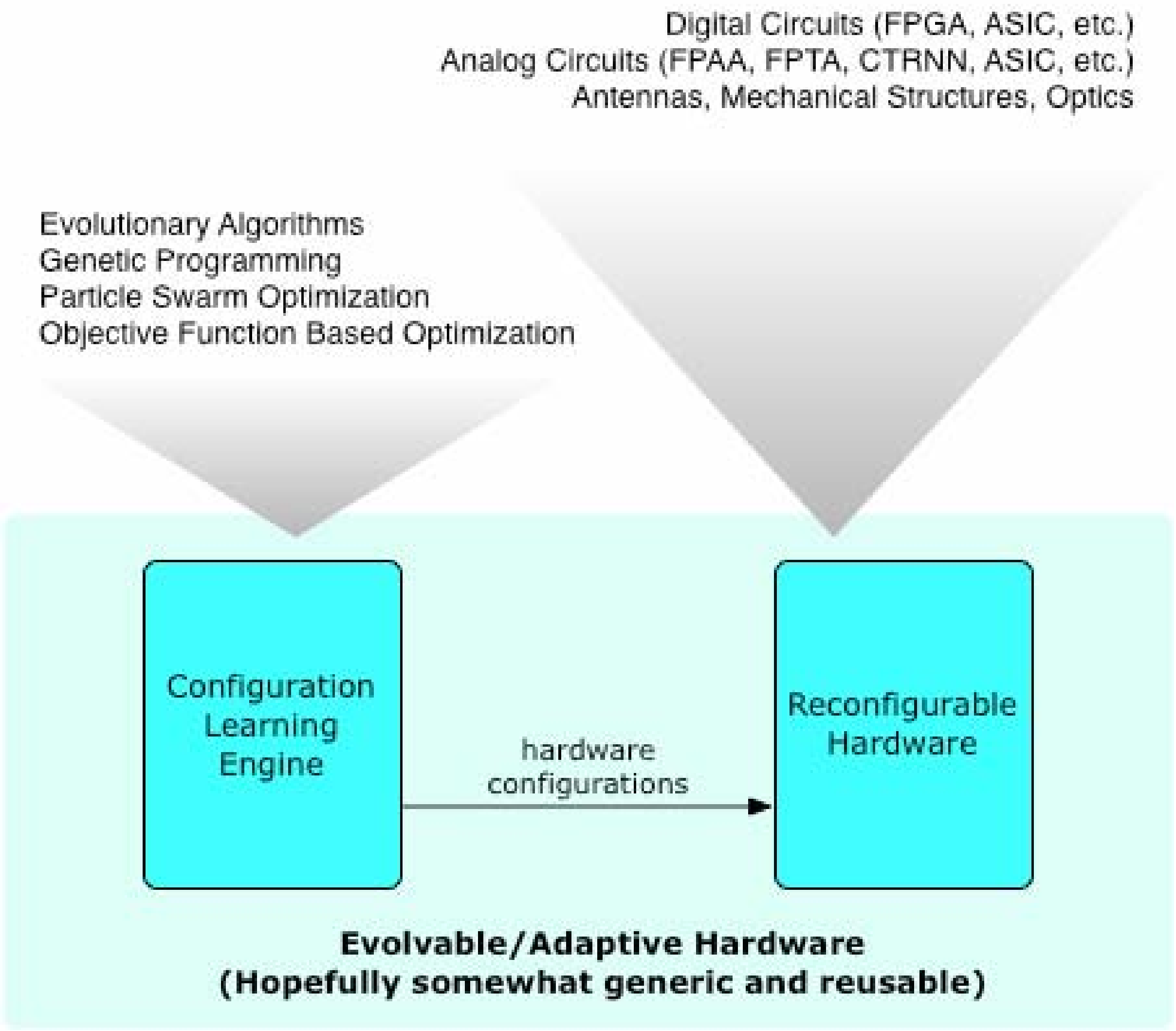


Evolvable and Adaptive Hardware (EAH)

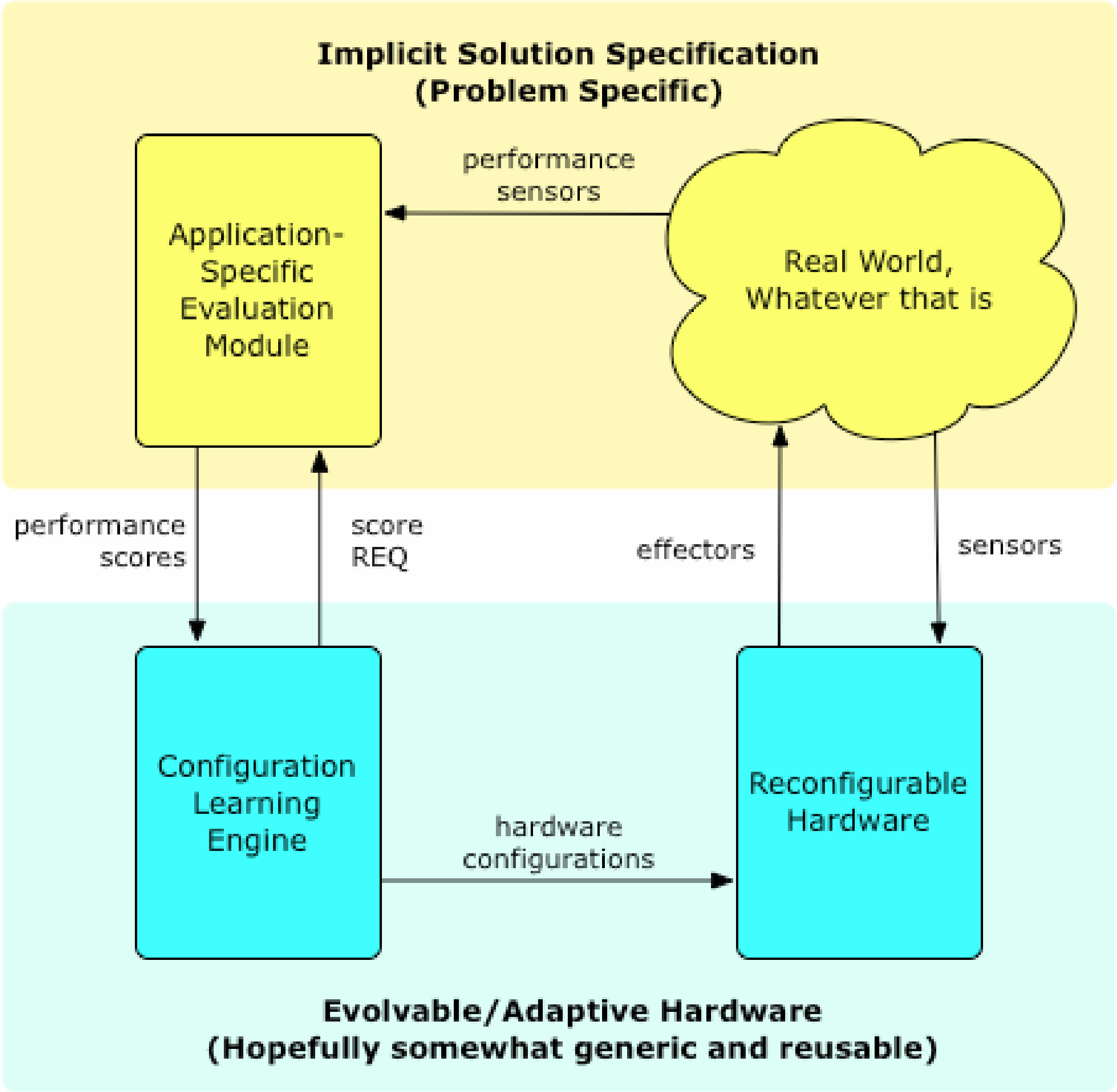


IEEE Computational Intelligence Society

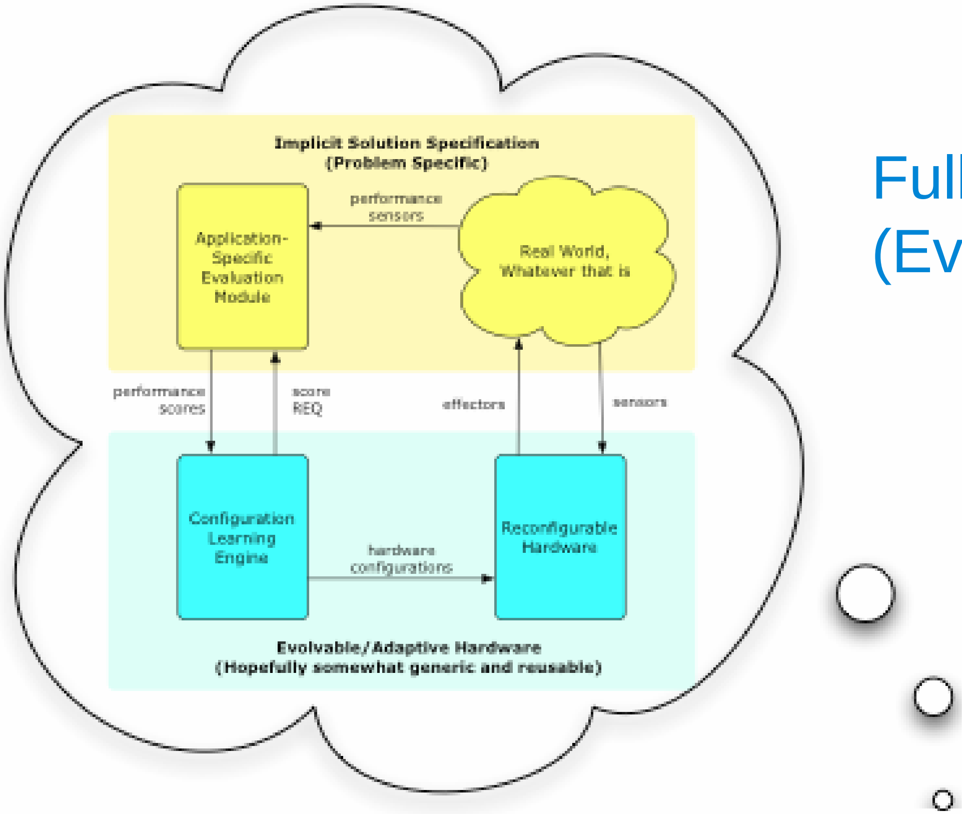
EAH: Search and Substrate Choices



Evolvable and Adaptive Hardware (EAH)

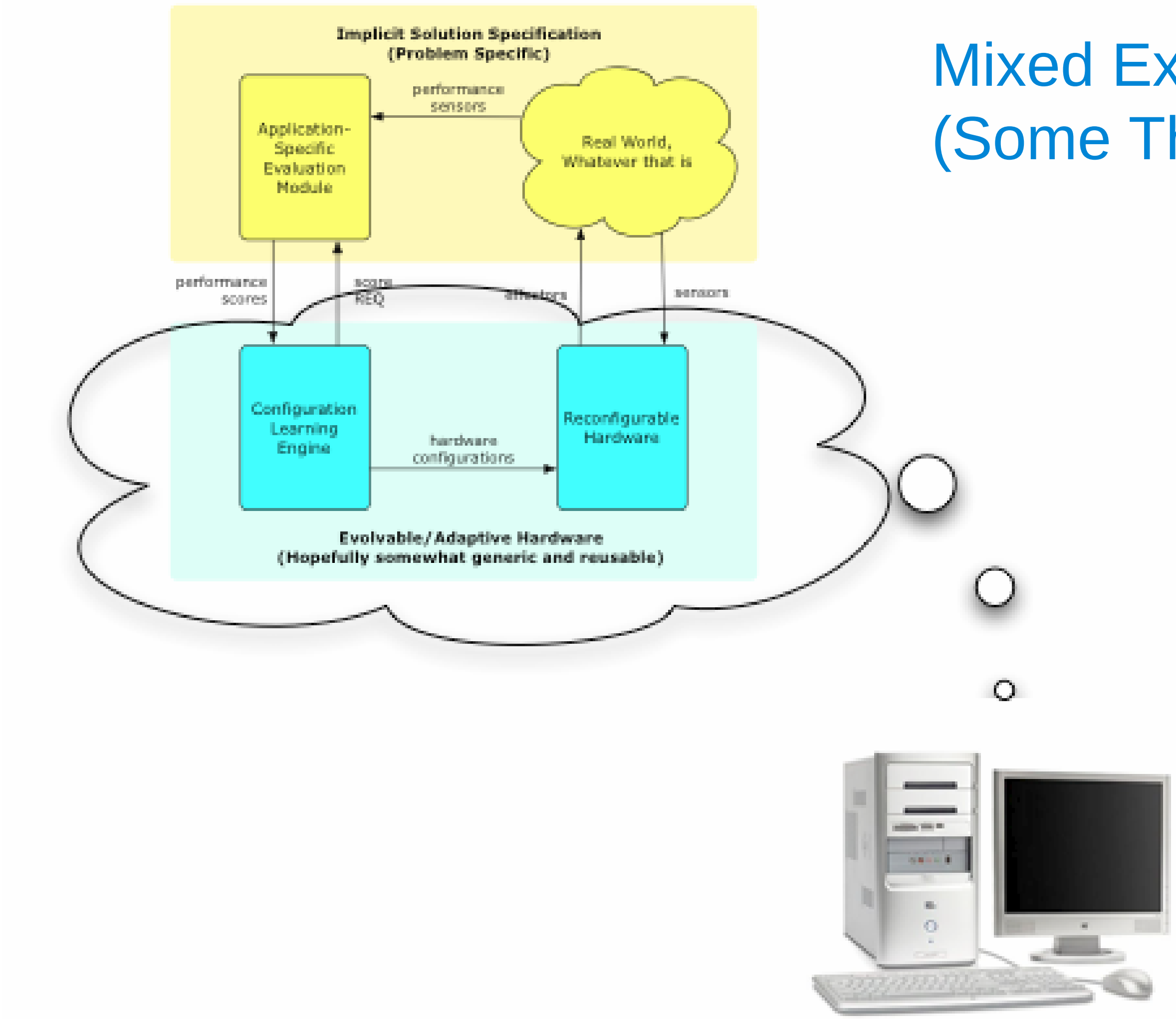


EAH: Simulated or Real?



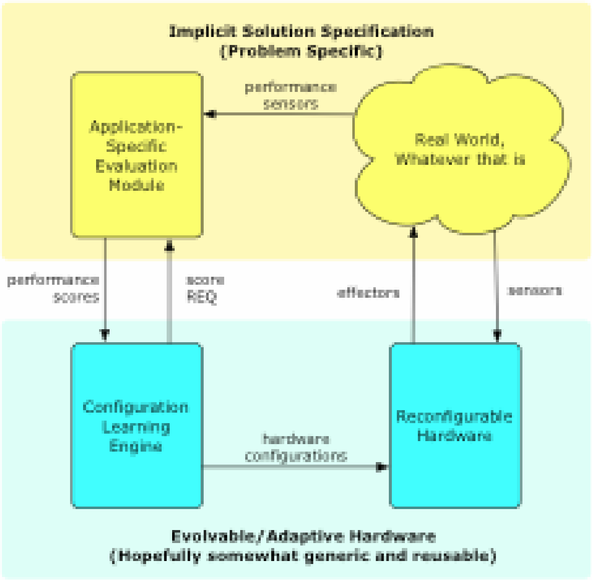
Fully Extrinsic
(Everything Simulated)

EAH: Simulated or Real?



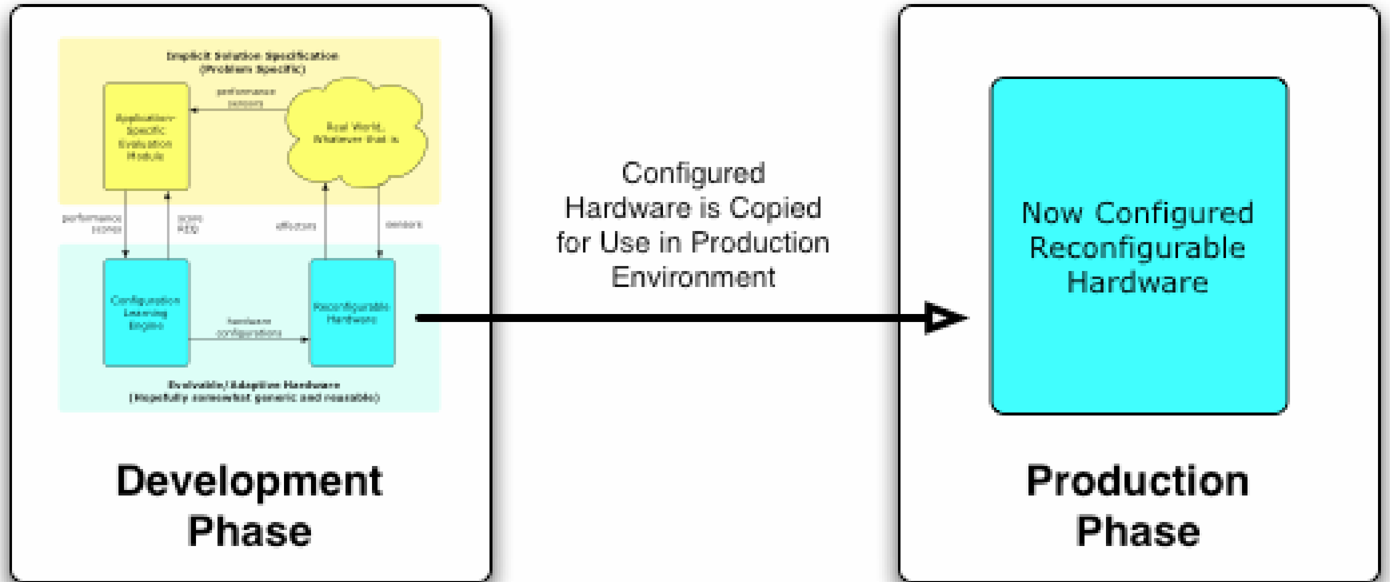
Mixed Extrinsic/Intrinsic
(Some Things Simulated)

EAH: Simulated or Real?



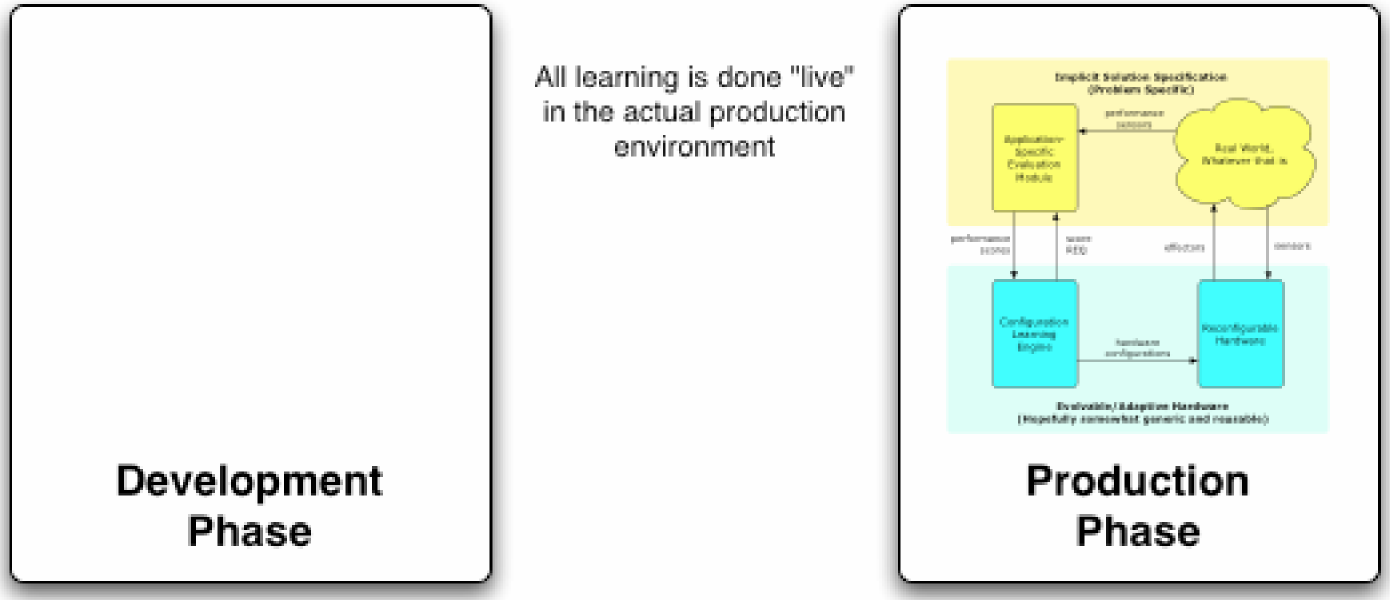
Fully Intrinsic
(Nothing Simulated)

EAH: Online or Offline?



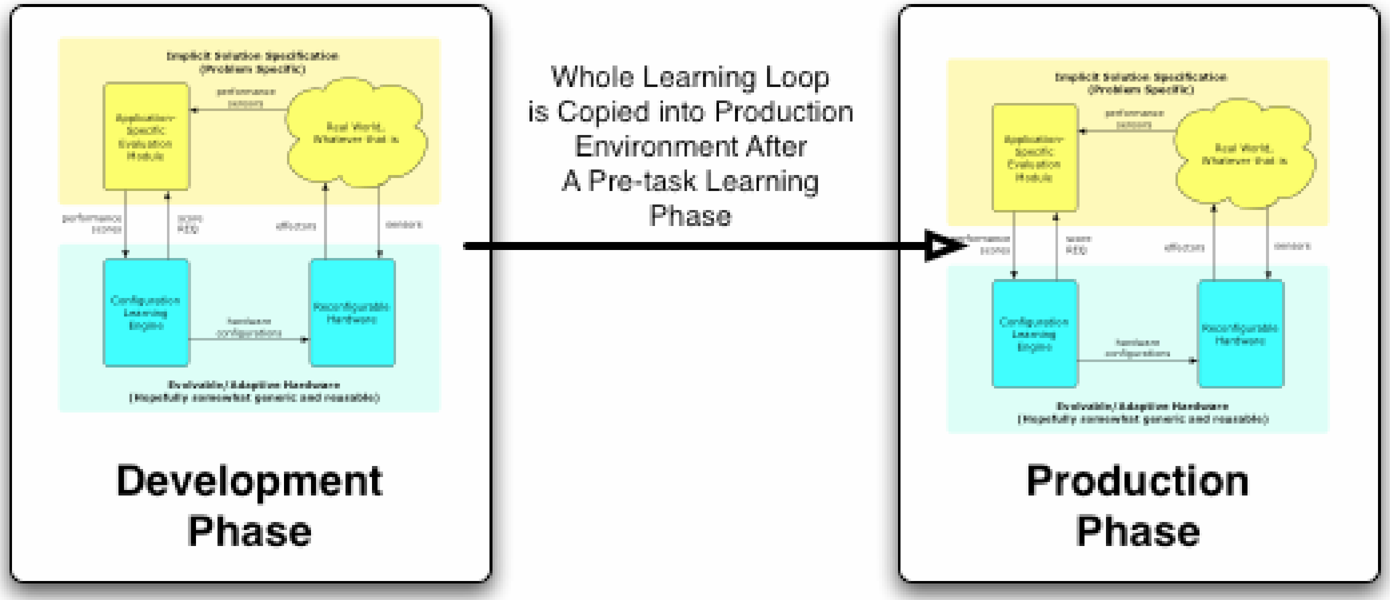
Offline Evolution

EAH: Online or Offline?



Online Evolution

EAH: Online or Offline?



Mixed Online/Offline Evolution

Basic Definitions: Recap and Summary

When	*How*	*What*
Online Offline Mixed O/O	Extrinsic Intrinsic Mixed E/I	Evolvable Hardware

Evolvable →

Evolutionary Algorithms
Genetic Programming
Particle Swarm Optimization

Other Objective Function
Based Optimization Methods

Hardware →

Digital Electrical Circuits
Analog Electrical Circuits
Optical Structures
Mechanical Structures
Antennas and Waveguides

Other confgurable/reconfigurable
hardware substrates

IEEE Computational Intelligence Society

Motivation: Why Do This at All?

- There seems to be two broad categories of motivations in the literature
  - To find/create/produce/grow devices and machines drawn from beyond the boundaries of current engineering practice that are somehow better than traditionally engineered alternatives
  - To create devices that are capable of autonomous self-modification for:
    - Self-repair in response to unexpected damage modalities
    - Self-improvement in taking advantage of unexpected opportunities
- Of course, mixes of the above two motivations are possible and often observed.

Motivation: Why Do This at All?

- To get a better handle on these motivations, it's useful to consider briefly what EH is *NOT*
- *EAH IS NOT DESIGN OPTIMIZATION*
  - Design Optimization uses optimization techniques to tune parameters of a design template. The template is chosen by a human and the tuning algorithm is not given the ability to transcend the template. E.G. No matter how much one tunes a linear proportional controller, it remains a linear proportional controller with properties and limitations known *apriori*.
  - EAH uses optimization techniques to generate and tune device configurations that transcend the boundaries of pre-determined design templates. *Many, if not all, hardware substrate / search method combinations observed in the literature are specifically designed to encourage out-of-the-box candidate solutions.*
- It is in the above context that both the benefits, and potential pitfalls, of EAH research can be seen.

Motivations: Online vs. Offline

- **Offline EAH** efforts are usually aimed at finding novel solutions to challenging problems. Note that this only possible if the EAH system can transcend pre-conceived boundaries on solution type.

- **Online EAH** efforts are usually aimed at exploiting unexpected opportunities or self-repair while a device is in service. Note too that these goals are well-served by EAH's flexibility.

- **Mixed Online/Offline** efforts often combine both motives. The offline portion finds a solution that is "in the ballpark". The online portion modifies that solution on the fly to the needs of the moment.

Motivations: Intrinsic vs. Extrinsic

- **Extrinsic EAH** efforts usually adopt simulation for one or more of the following reasons:
  - Speedup: For some systems, a simulation may allow many more candidates to be considered per tick of wall clock time. More candidates considered can translate into better search results.
  - Safety: Some candidate configurations might be “too dangerous” to try in real hardware. Simulation is a safe alternative.
  - Expense: Simulation can be more inexpensive than constructing (and possibly breaking) real hardware.
  - Others? Anyone?

Motivations: Intrinsic vs. Extrinsic

- **Intrinsic EAH** efforts usually employ real hardware for one or more of the following reasons:
  - Speedup: For some systems, the real thing might actually be faster than a simulation of it. This is particularly true with simulation of complex physical processes.
  - Model Fidelity: Some systems can be very difficult to simulate with sufficient fidelity that a EA process won't find and exploit simulation artifacts. Forgoing the simulation is one way to avoid that problem.
  - Others? Anyone?

Motivations: Recap and Summary

When	*How*	*What*
Online Offline Mixed O/O	Extrinsic Intrinsic Mixed E/I	Evolvable Hardware

Evolutionary Algorithms
Genetic Programming
Particle Swarm Optimization

Other Objective Function
Based Optimization Methods

Digital Electrical Circuits
Analog Electrical Circuits
Optical Structures
Mechanical Structures
Antennas and Waveguides

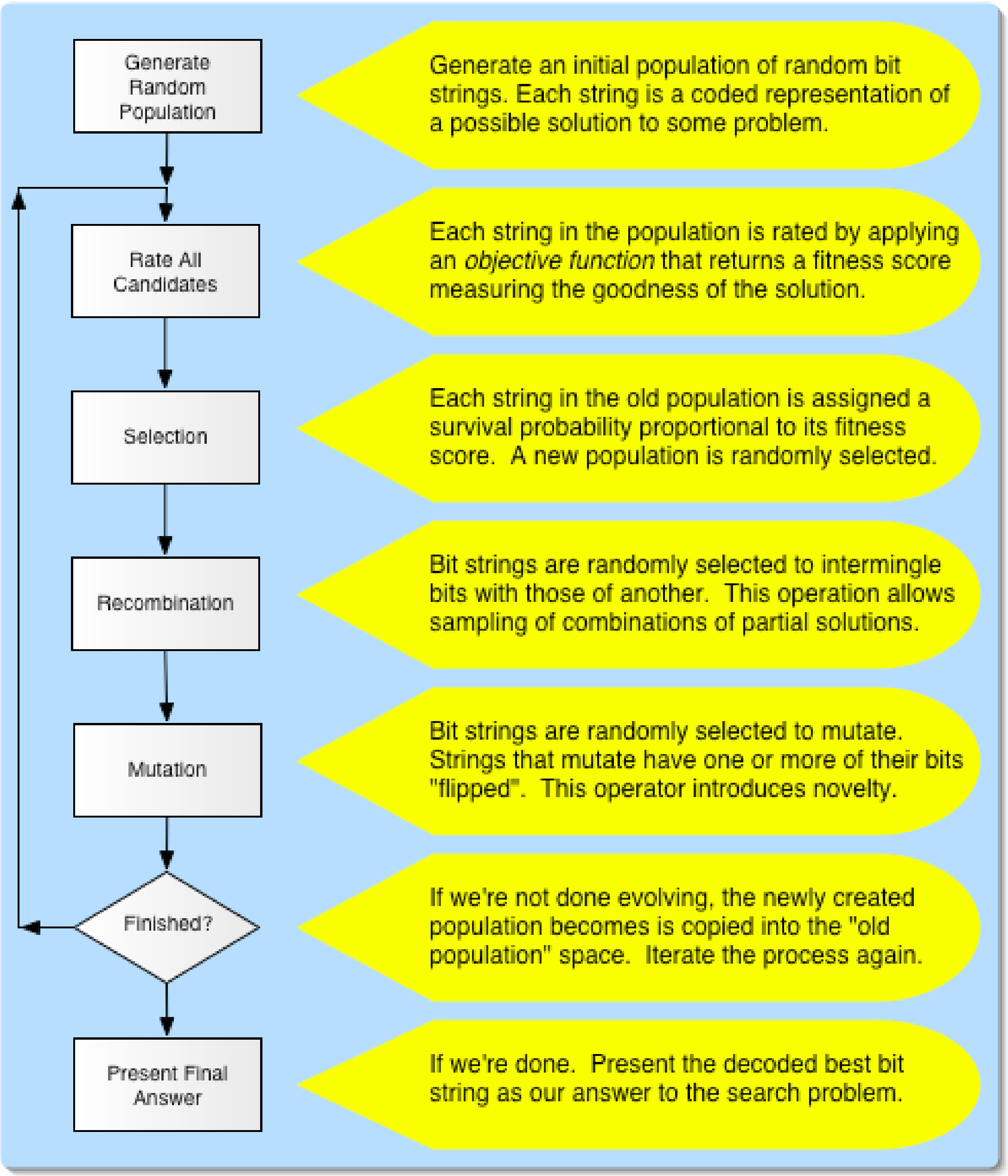
Other confgurable/reconfigurable
hardware substrates

Basic Techniques

- The two most basic choices that need to be made are “which search/learning engine” and “what reconfigurable hardware substrate”.
- Naturally, the “best choice” for a particular application can and will be problem dependant.
- Further, practitioners can and do differ in their opinions with regard to both choices.
- Many issues with regard to efficacy and appropriateness are still open issues under investigation. (Insert shameless WEAH plug here)

Basic Techniques: Search Algorithms

- Something is needed to generate, test, and improve hardware configurations with respect to a user-defined objective function. We'll generically refer to that as the "search algorithm"
- Within EAH, Evolutionary Algorithms are particularly popular, but they are not the only possible choice. Anything that optimizes a description of the hardware configuration based on a scalar objective function could in principle work.
- Let's first consider a generic Genetic Algorithm (The experts can take a coffee break here)



IEEE Computational Intelligence Society

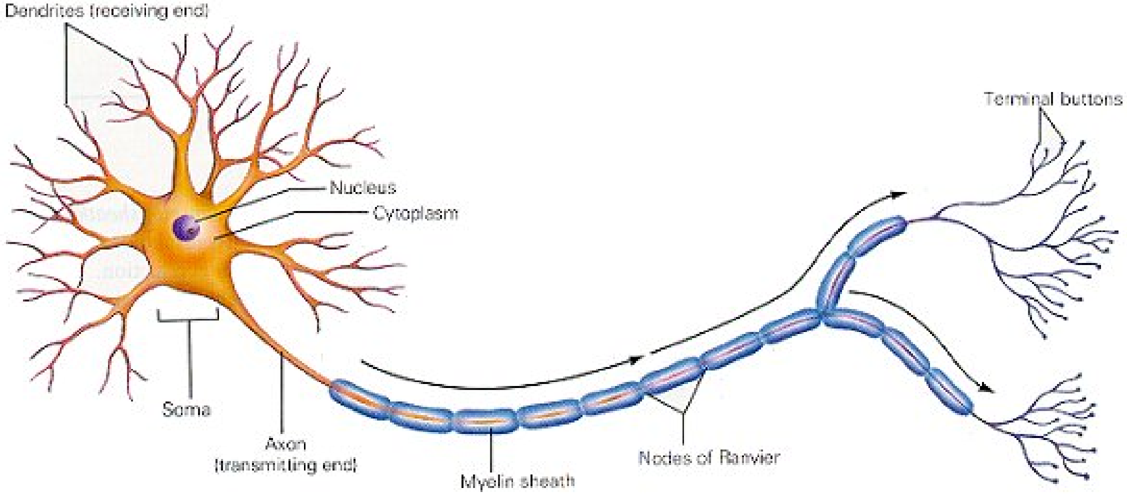
Basic Techniques: Search Algorithms

- In many ways, the very same issues that concern the EAH community concern the Evolutionary Computation community in general. Two of these issues are:
  - Search efficacy and speed
  - Ability to optimize in dynamic and noisy environments
  - Online vs. Offline performance
- For EAH practitioners, the following issues may be important in addition to the above:
  - Size of hardware implementation
  - Exclusion of “dangerous” candidates from population
- A wide variety of EA types have been used. A full survey is beyond the scope of this tutorial. One may, however, track down examples in the references appearing at the end of these slides.

Basic Techniques: Hardware Substrates

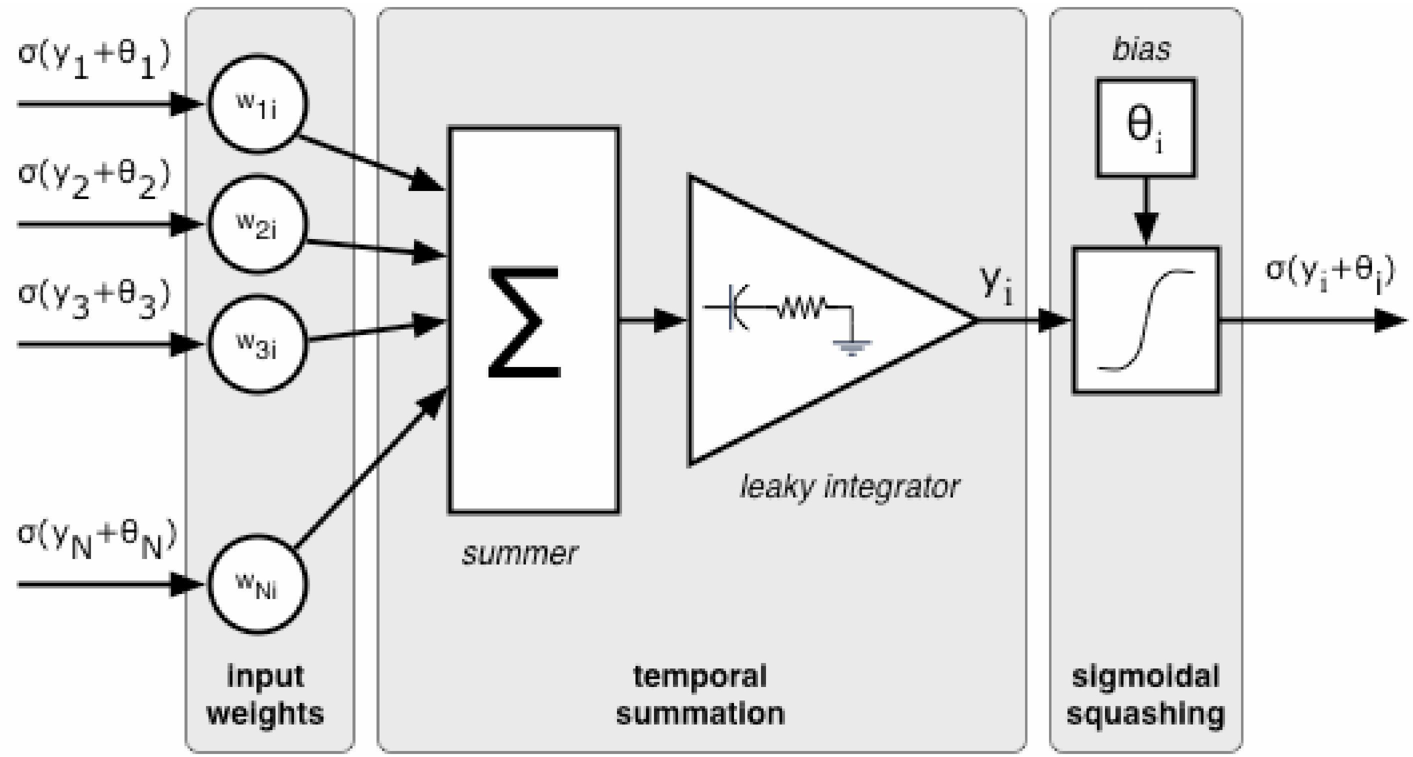
- There is huge variation across the community as to what constitute appropriate substrates. Some variation is problem domain dependant. Some variation is due to differences in opinion and/or matters of practicality and economy.
- We will not attempt a full survey of all possibilities. We will discuss what characteristics are shared by “good substrates”. Then we will examine some notable examples of substrates in the literature.

Basic Techniques: Case Study



IEEE Computational Intelligence Society

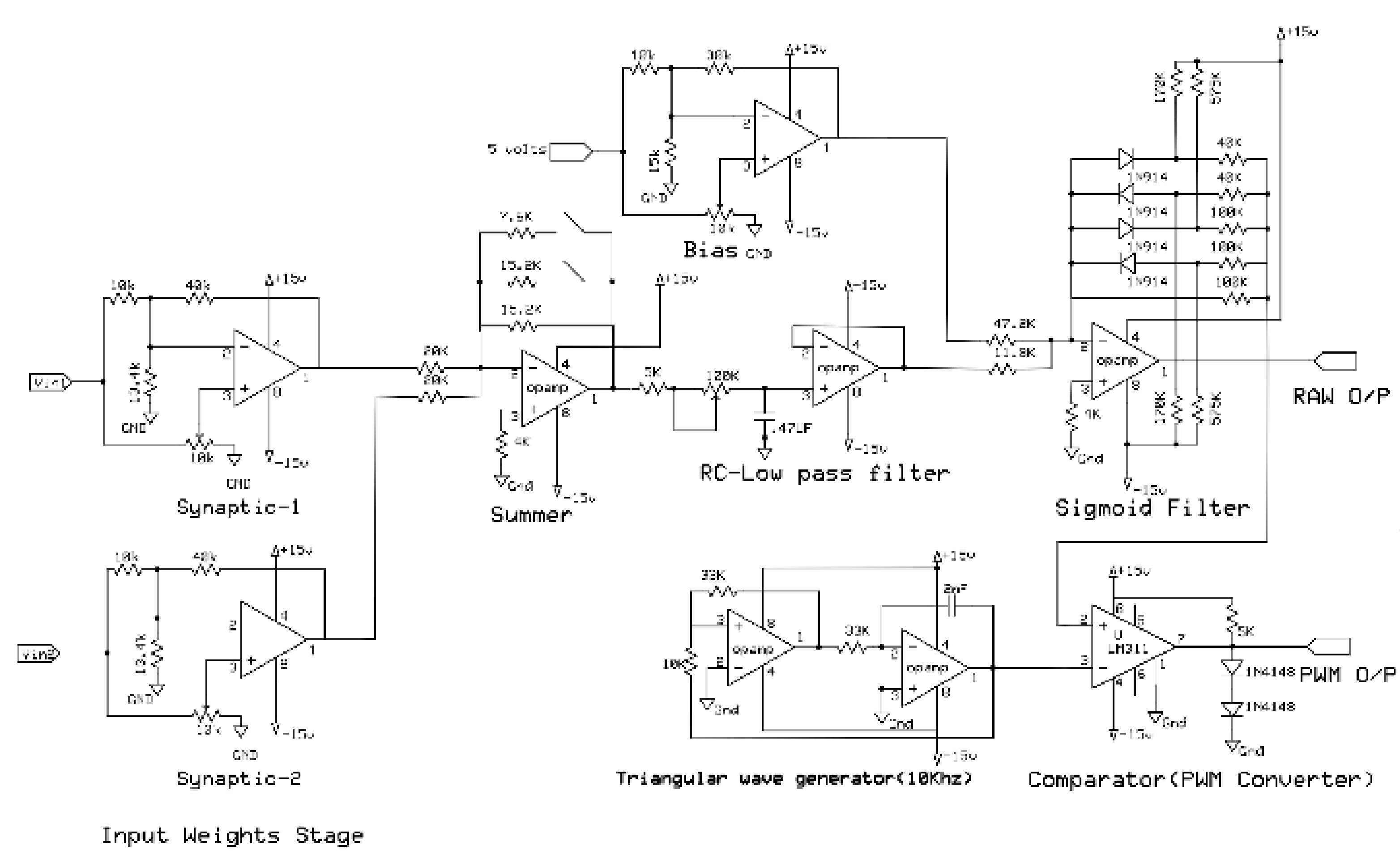
Basic Techniques: Case Study



$$\tau_i \frac{dy_i}{dt} = -y_i + \sum_{j=1}^{N} w_{ji}\sigma\left(y_j + \theta_j\right) + s_i I_i(t) \qquad i = 1, \text{☺}, N$$

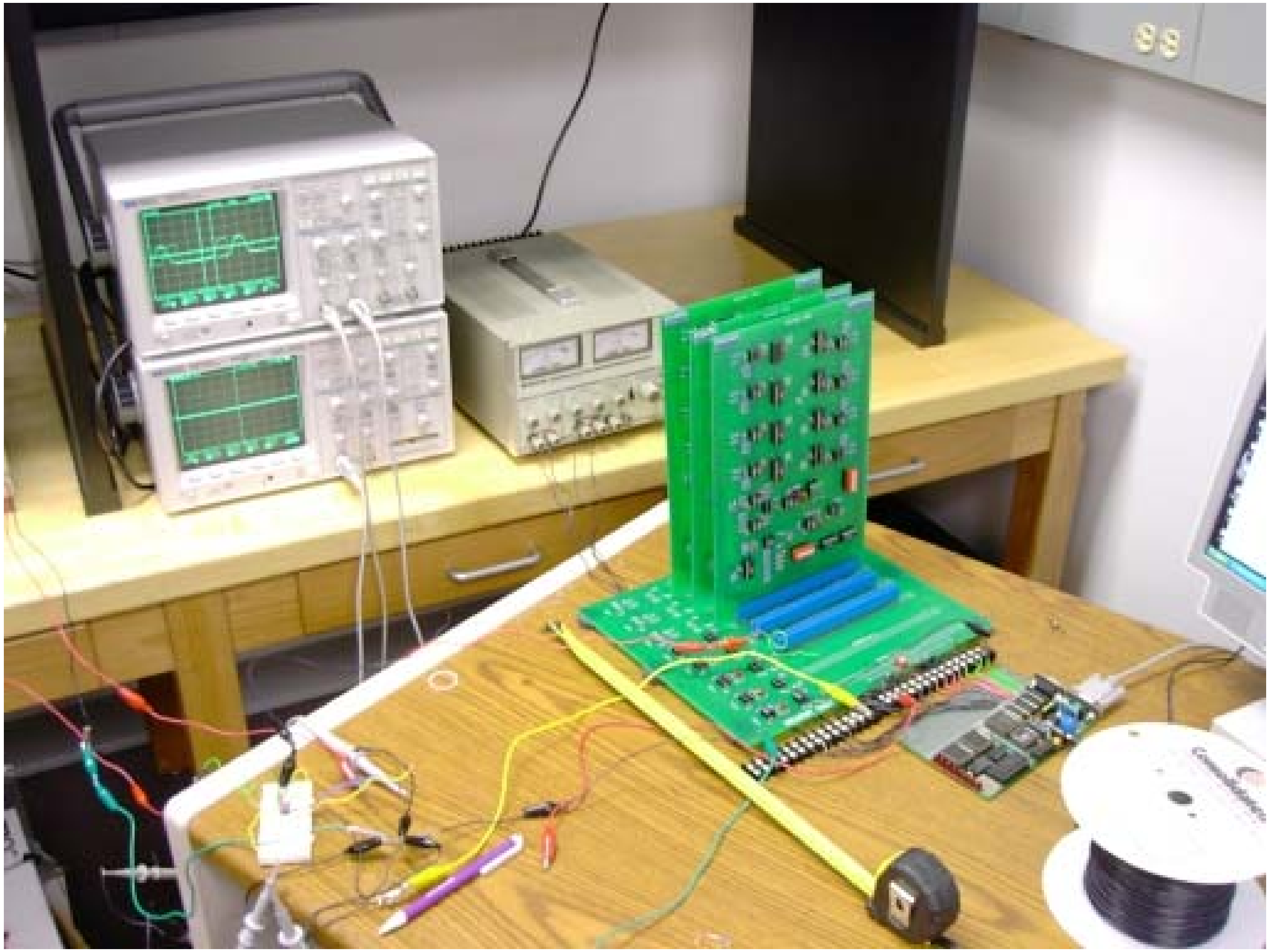
IEEE Computational Intelligence Society

Basic Techniques: Case Study



IEEE Computational Intelligence Society

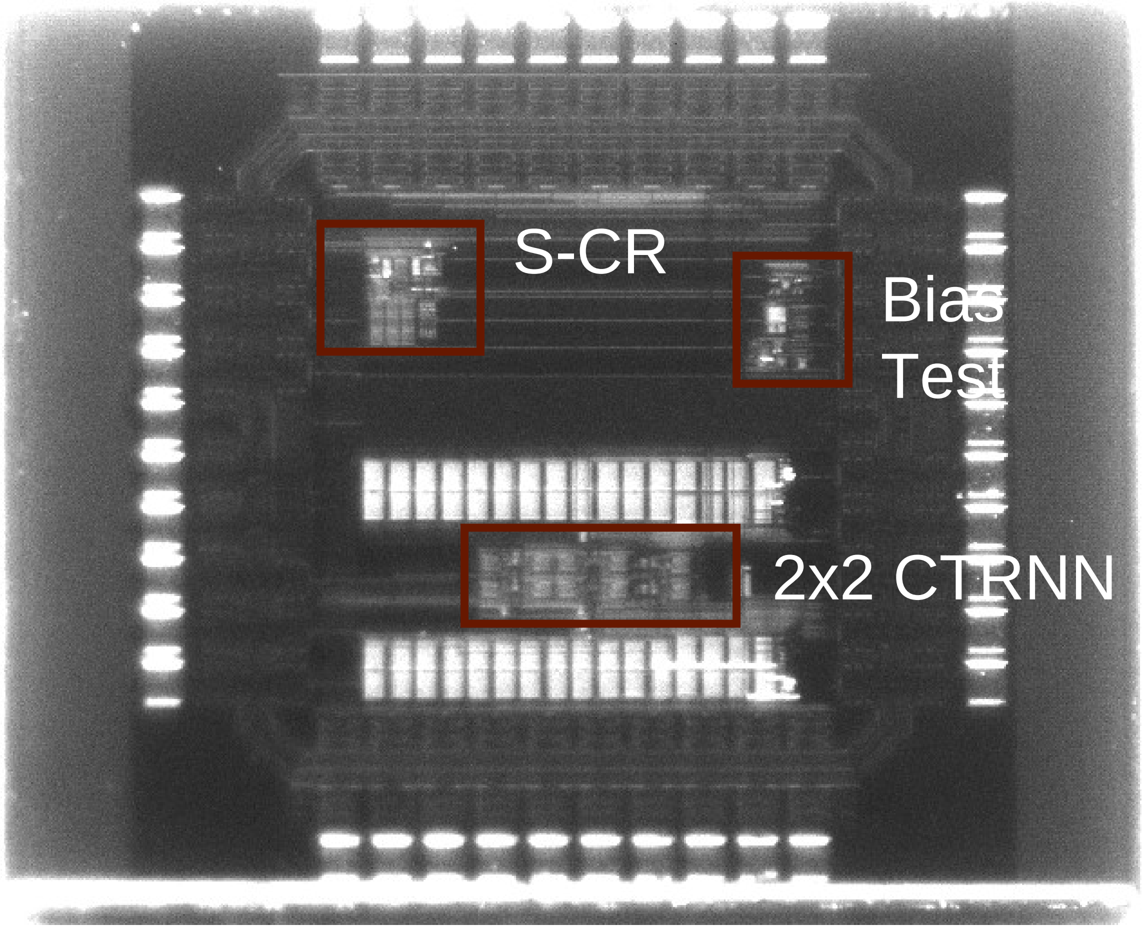
Basic Techniques: Case Study



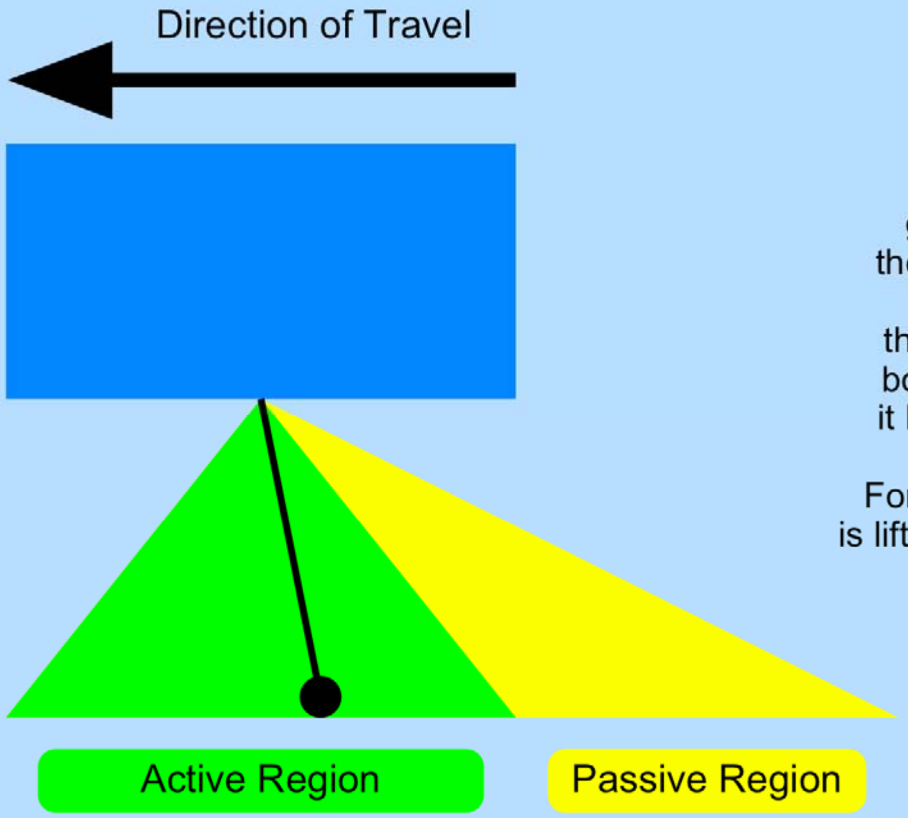
IEEE Computational Intelligence Society

Basic Techniques: Case Study

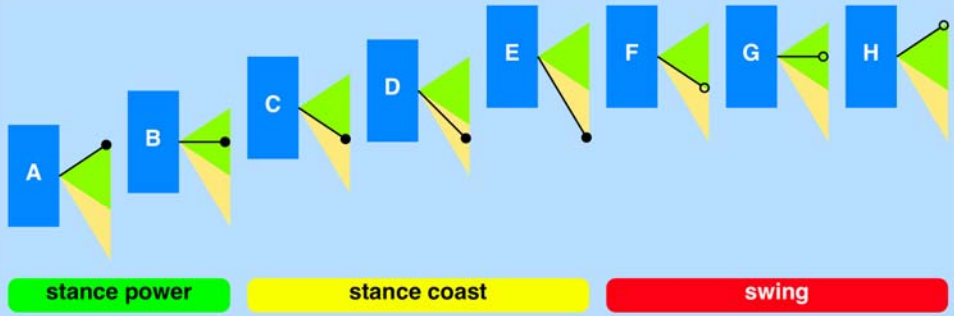
1.5 x1.5 mm



Basic Techniques: Case Study



The agent moves by "rowing" with its single leg. The leg swings about its joint with the body to the full forward position (to the far left of the green active region). It plants its foot there, and stances the leg, pushing the body forward. Force is applied until the leg enters the passive region. The body then coast with whatever velocity it had attained until the leg reaches the rightmost limit of the passive region. Forward motion then stops until the foot is lifted and the leg swung back to the left edge of the active region -- at which time a new step can start.



A: Begin Power Stance Phase
Leg is fully forward, foot is planted. Backward Swing (BS) control signal should be fully on and Forward Swing control signal should be fully off so that maximal torque, and thus maximal translational acceleration, is applied to the body.

B: Middle Power Stance Phase
Body has translated forward and is still accelerating. Maximal BS and Minimal FS should be applied.

C: Begin Coast Stance Phase
Body has translated far enough that the leg angle has stretched into the passive (hyperextended) region. The body will continue to translate with whatever speed it had already achieved while the leg is in this region. This models the reduced ability of a hyperextended limp to provide propulsion forces.

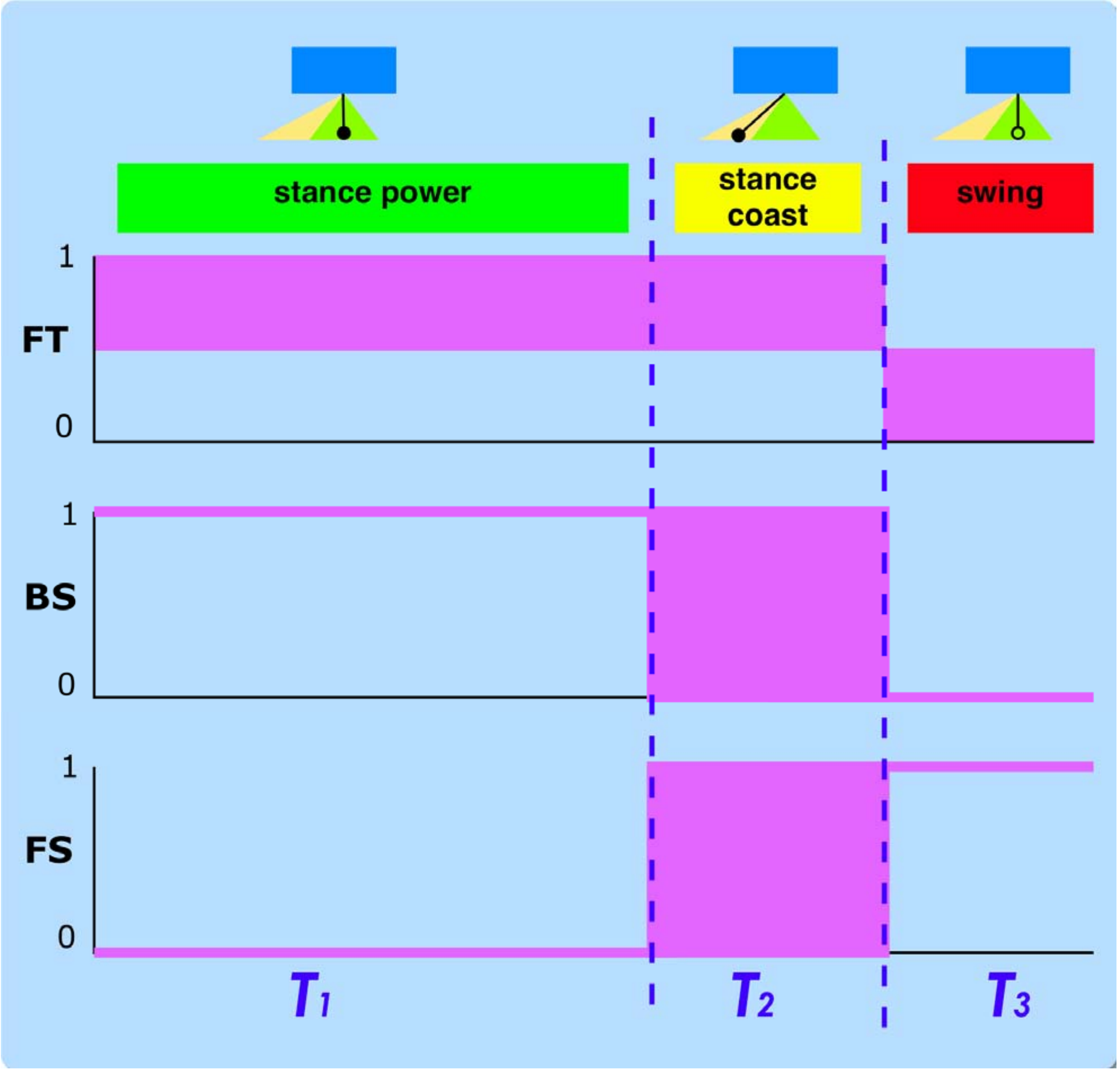
D: Middle Coast Stance Phase
The body has coasted forward further. Note that so long as the foot remains planted, it doesn't matter what control signals are applied for BS and FS, as they can not affect the agent.

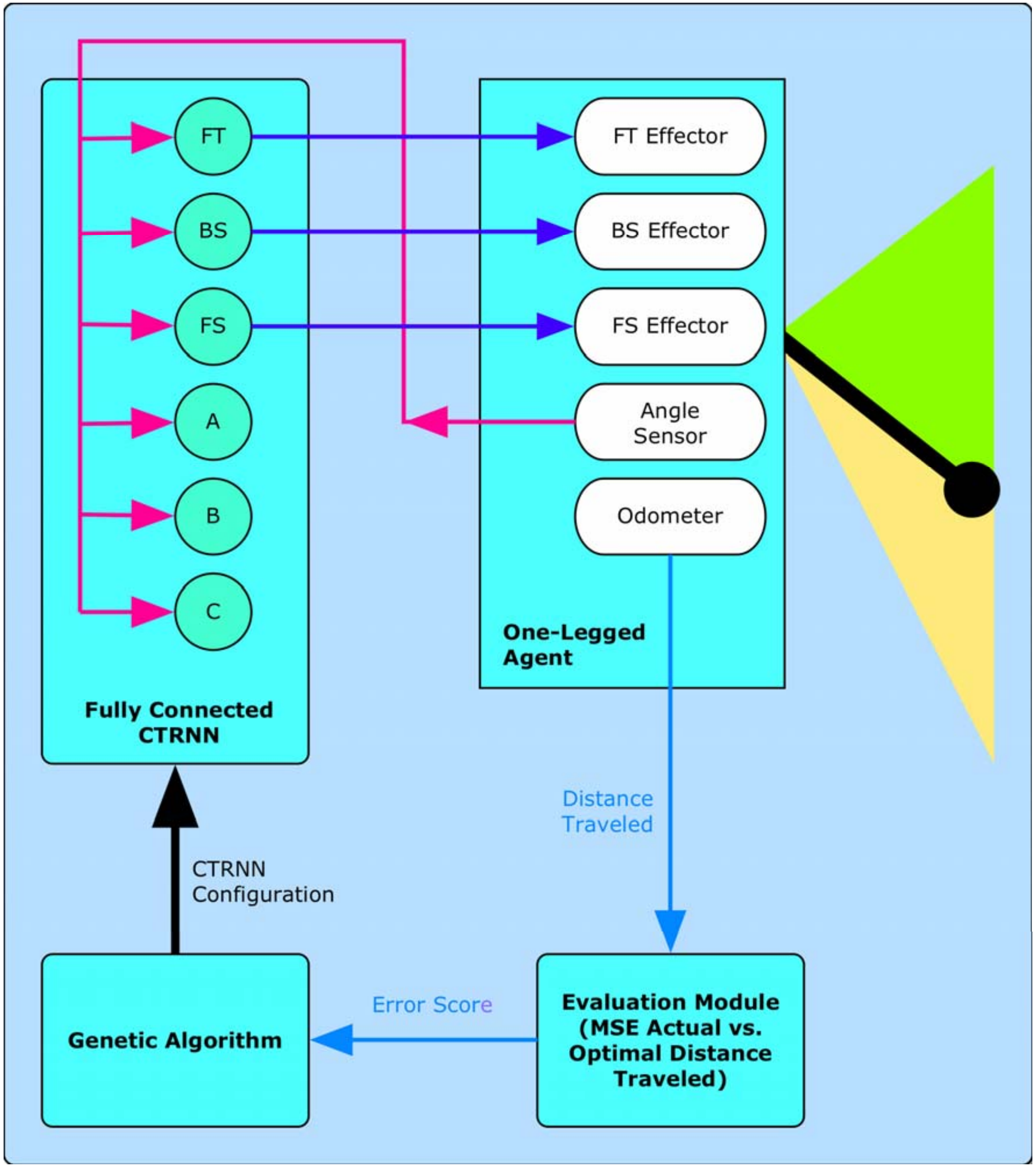
E: End Coast Stance Phase
The body has coasted far enough that the leg has stretched to the end of the passive region. The leg can be considered to have hit a mechanical stop. So long as the foot remains planted, it acts as an anchor and no further forward motion is possible.

F: Begin Swing Phase
The foot is lifted (open circle). The leg immediately snaps back to the limit of the active range. The BS control signal should be full off and the FS control signal should be full on to command maximum angular acceleration about the leg joint to quickly return the leg to the full forward position.

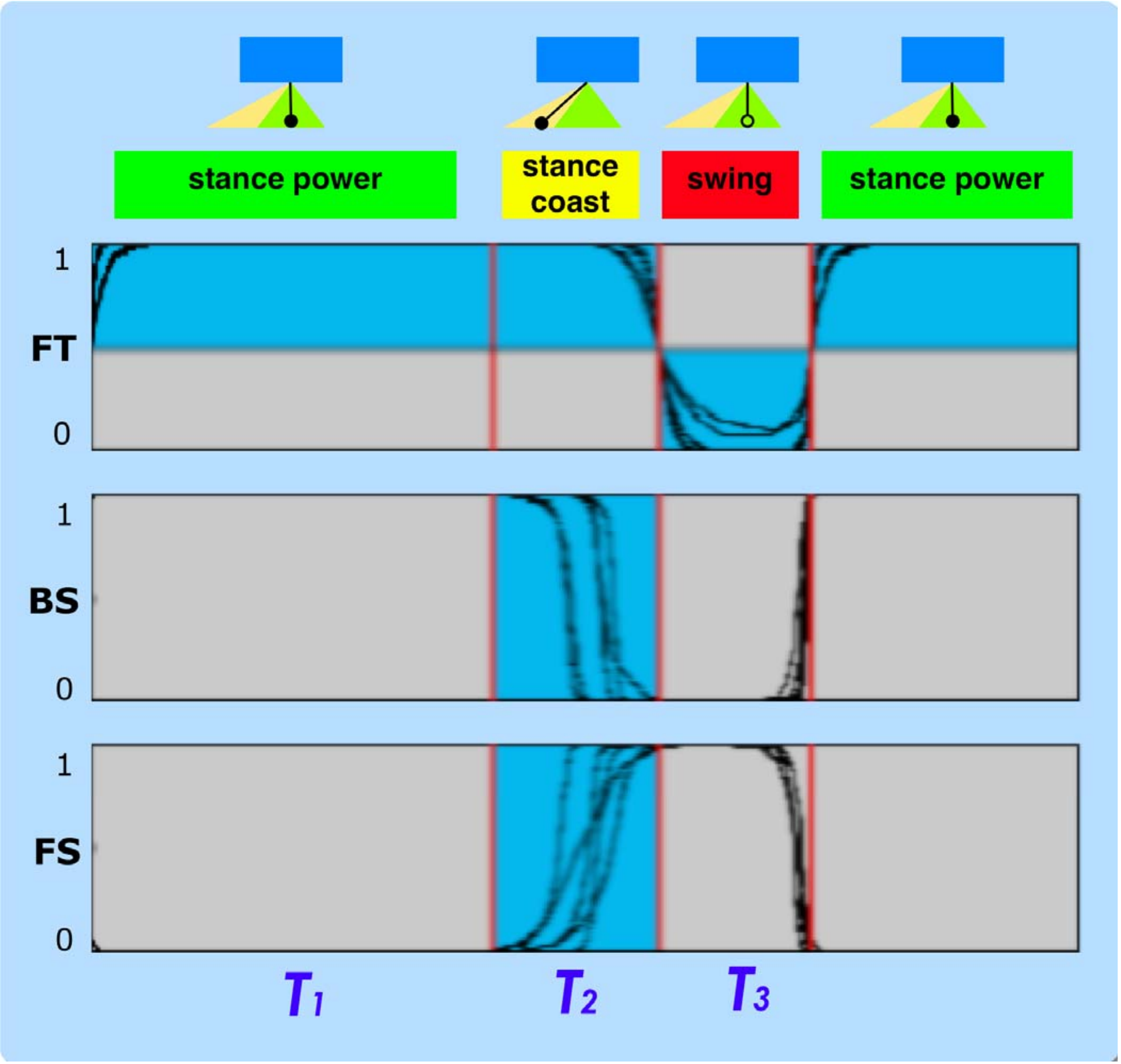
G: Middle Swing Phase
The leg is midway to returning to its full forward position.

H: End Swing Phase
The leg has returned to its full forward position. Once the leg effector is commanded to plant, then a new step can begin (return to agent configuration A)


stance power
stance coast
swing
1
FT
0
1
BS
0
1
FS
0
T1
T2
T3

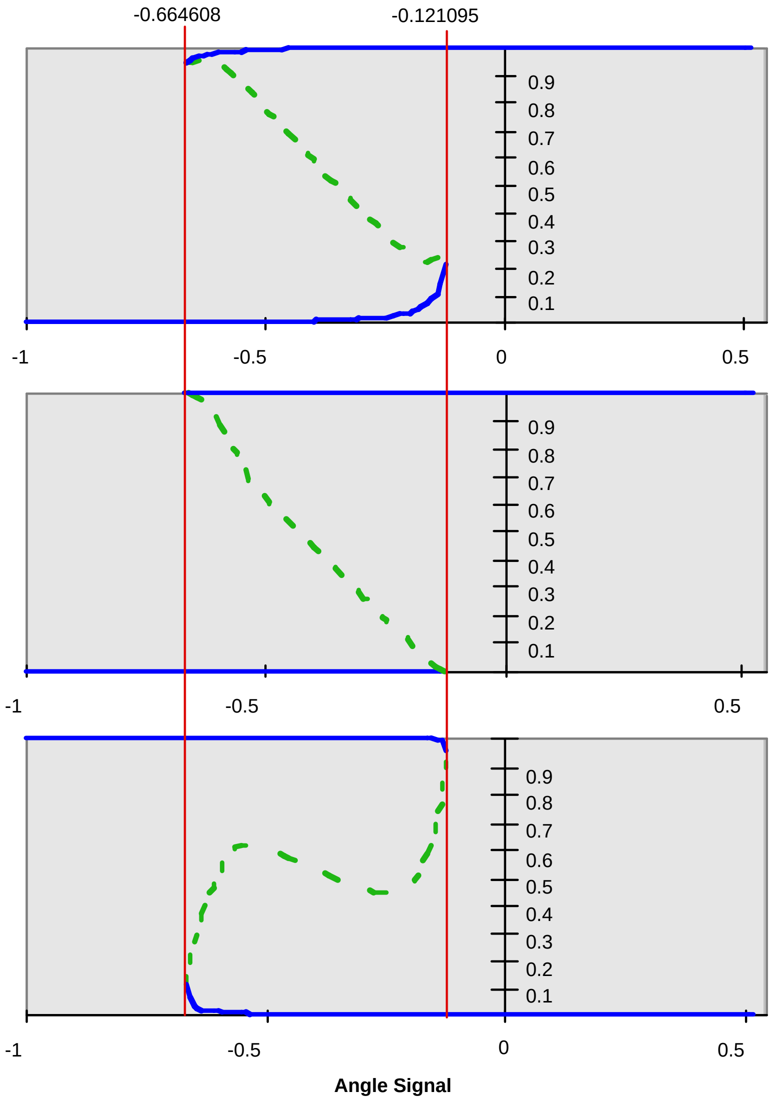


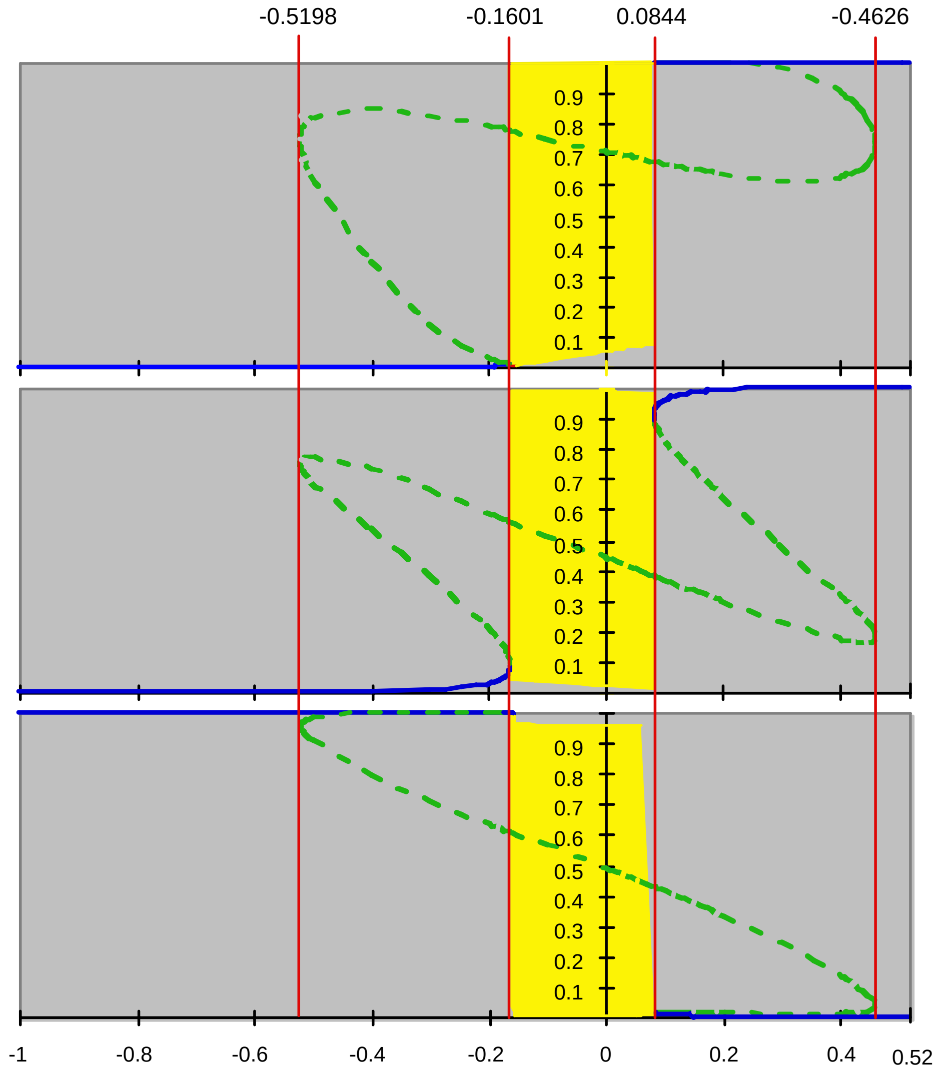
FT
BS
FS
A
B
C
Fully Connected CTRNN
FT Effector
BS Effector
FS Effector
Angle Sensor
Odometer
One-Legged Agent
Distance Traveled
CTRNN Configuration
Genetic Algorithm
Error Score
Evaluation Module (MSE Actual vs. Optimal Distance Traveled)

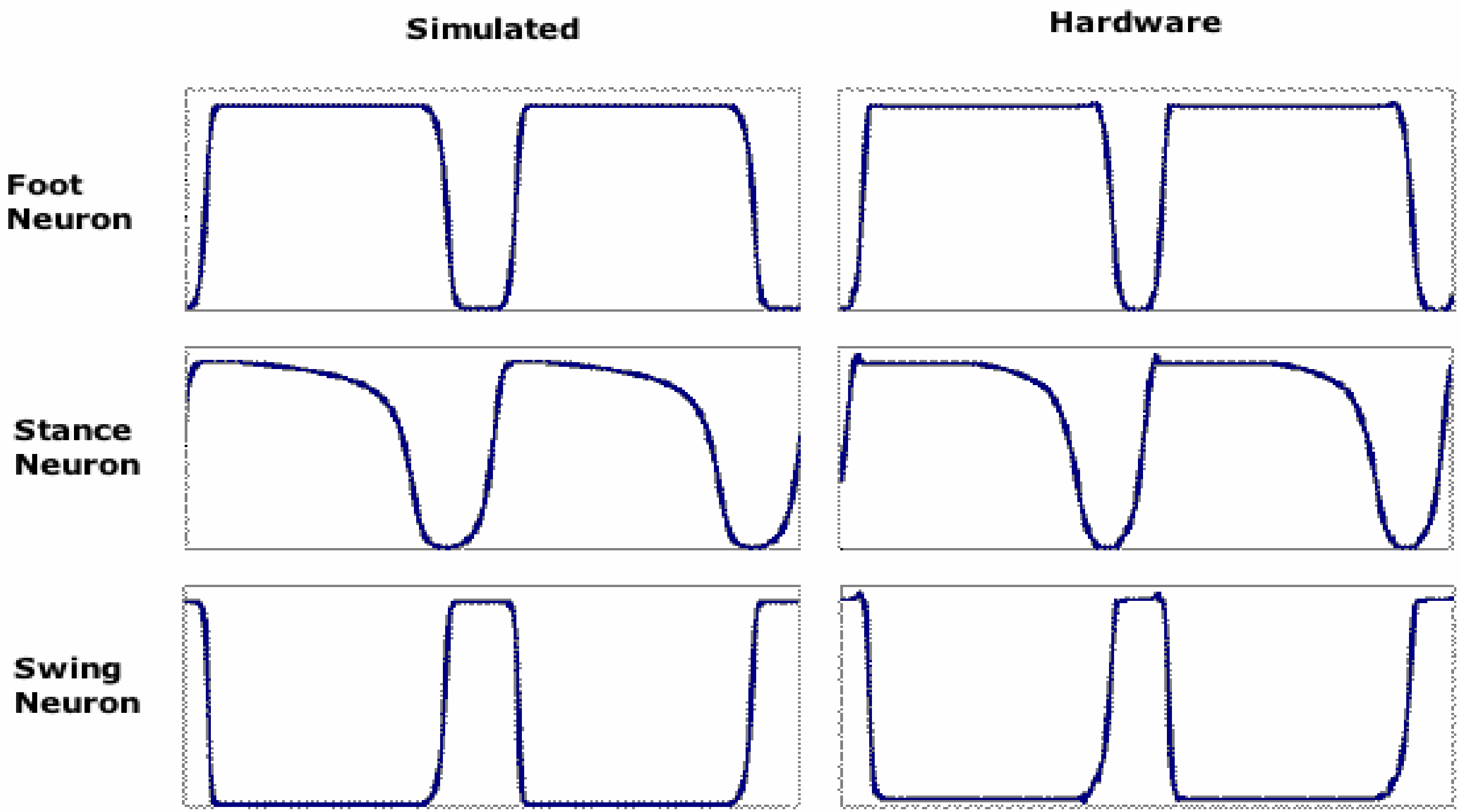


IEEE Computational Intelligence Society

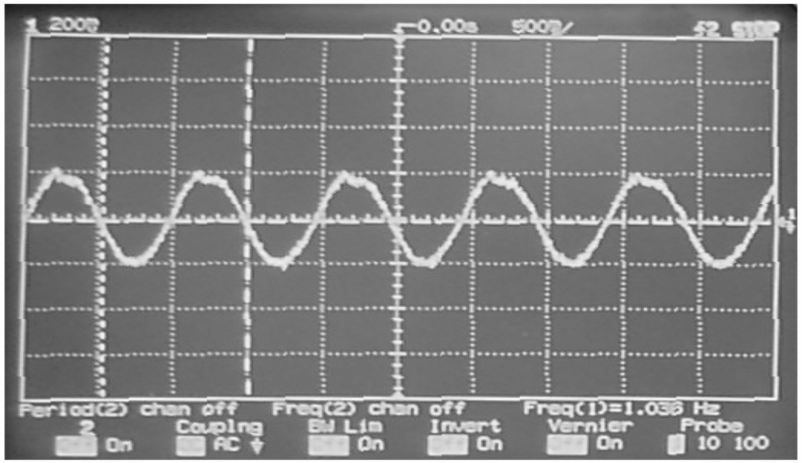
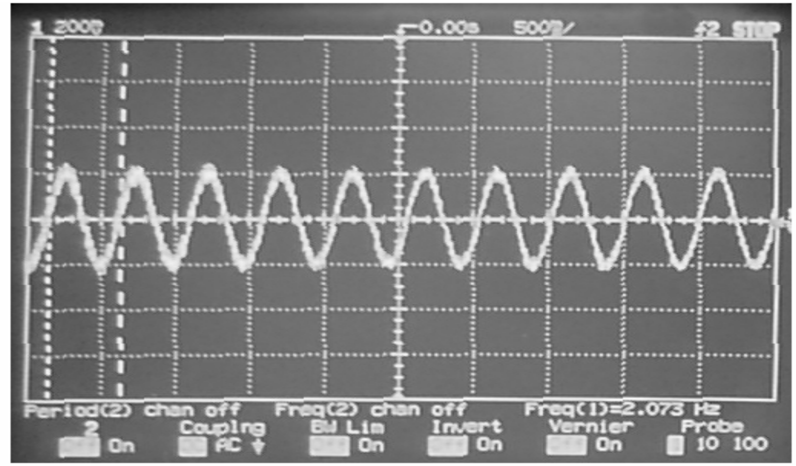
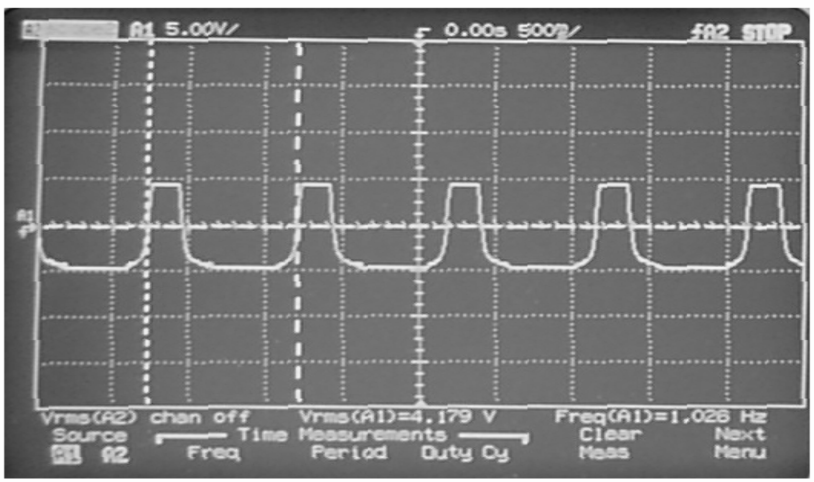
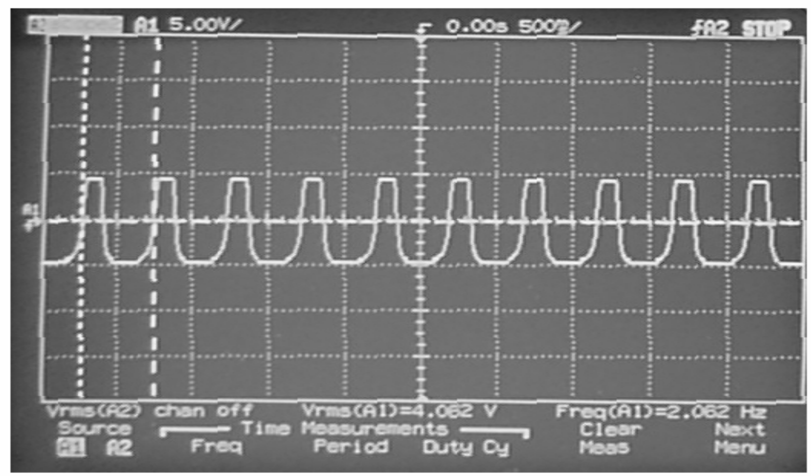
QuickTime?and a
Graphics decompressor
are needed to see this picture.


-0.664608
-0.121095
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
-1
-0.5
0
0.5
Angle Signal

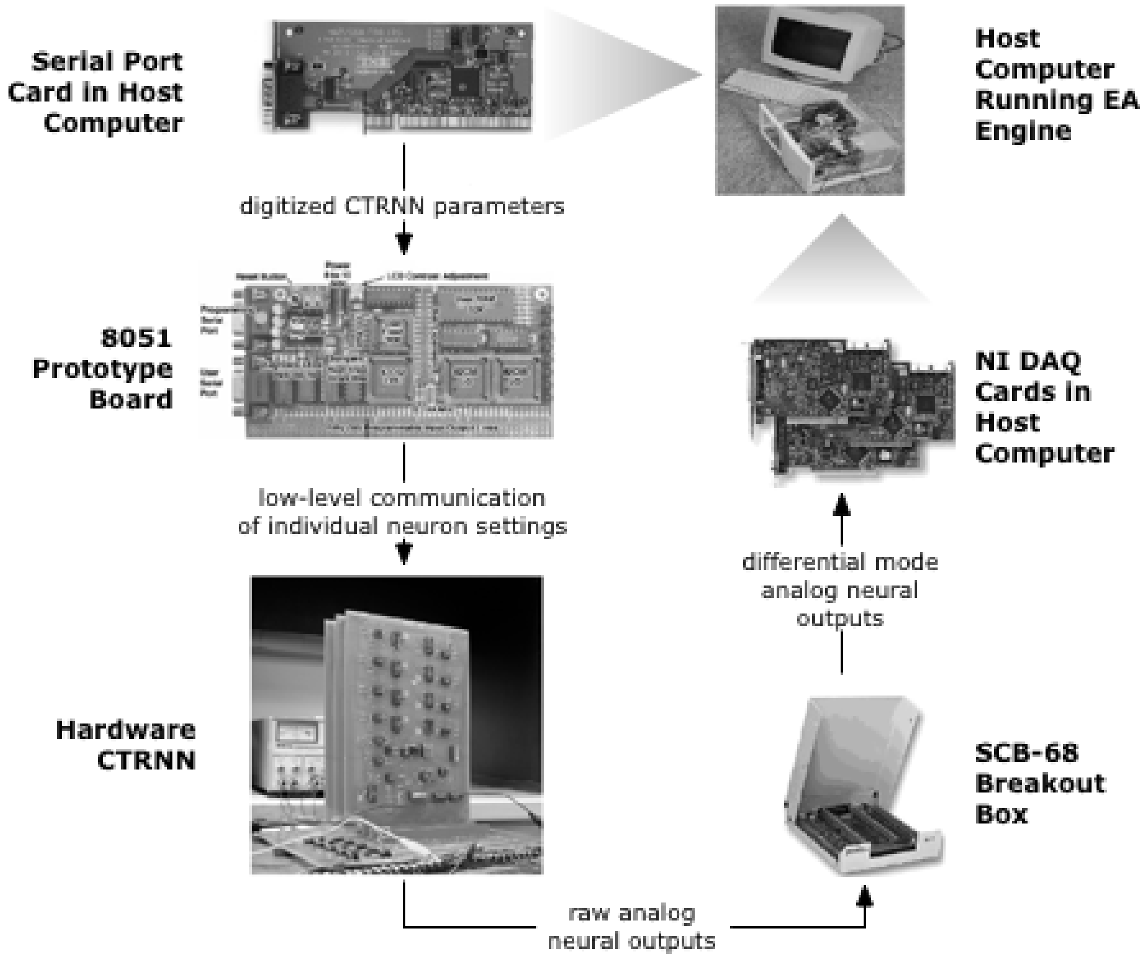

-0.5198
-0.1601
0.0844
-0.4626
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.52



IEEE Computational Intelligence Society


0.00s
f2 STOP
Period(2) chan off
Freq(2) chan off
Freq(1)=1.036 Hz
2
On
Coupling
AC
BW Lim
On
Invert
On
Vernier
On
Probe
10 100

0.00s
f2 STOP
Period(2) chan off
Freq(2) chan off
Freq(1)=2.073 Hz
2
On
Coupling
AC
BW Lim
On
Invert
On
Vernier
On
Probe
10 100

A1 5.00V/
0.00s
fA2 STOP
Vrms(A2) chan off
Vrms(A1)=4.179 V
Freq(A1)=1.026 Hz
Source
A1 A2
Time Measurements
Freq
Period
Duty Cy
Clear
Meas
Next
Menu

A1 5.00V/
0.00s
fA2 STOP
Vrms(A2) chan off
Vrms(A1)=4.062 V
Freq(A1)=2.062 Hz
Source
A1 A2
Time Measurements
Freq
Period
Duty Cy
Clear
Meas
Next
Menu

Basic Techniques: Case Study



IEEE Computational Intelligence Society

Basic Techniques: Case Study

	Shape		***Phase*** *(PErms)* *Units:time steps*	***Frequency*** *(Fcorre%)*	***Shape***		
	Magnitude *(MErms)* *Units:normalized magnitude*	***Slope*** *(SErms)* *No units*			***Magnitude*** *MeanAbsolute error*	***Slope*** *MeanAbsolute error*	***Phase*** *MeanAbsolute error*
Configuration 1	0.128169026	0.111669613	0	100%	0.0647	0.0589	0
Configuration 2	0.044774068	0.02366666	0	100%	0.0218	0.0133	0
Configuration 3	0.08662832	0.048784967	1	100%	0.0494	0.0275	1
Configuration 4	0.092819065	0.047603614	0.707106781	100%	0.0415	0.0225	0.5
Configuration 5	0.041369648	0.0331614	0	100%	0.0292	0.0192	0
Configuration 6	0.143986354	0.087264284	0	100%	0.0768	0.0433	0
Configuration 7	0.081610409	0.021087732	1	100%	0.0311	0.013	1
Configuration 8	0.066450743	0.046041062	0	100%	0.035	0.0236	0
Configuration 9	0.114596221	0.111884257	0	100%	0.0596	0.0558	0
Configuration 10	0.145185999	0.147954733	0	100%	0.0636	0.029	0
Configuration 11	0.11092404	0.050229117	0	100%	0.0635	0.0448	0
Configuration 12	0.052022812	0.037297799	0	100%	0.028	0.0171	0
Configuration 13	0.088586209	0.018852471	0	100%	0.0494	0.0097	0
Configuration 14	0.076716118	0.050438799	0	100%	0.0354	0.0212	0
Configuration 15	0.08481056	0.084804777	0	100%	0.048	0.0444	0
Configuration 16	0.137617621	0.069708659	0	100%	0.0438	0.0304	0
Configuration 17	0.082622534	0.024653691	0	100%	0.0344	0.014	0
Configuration 18	0.08561056	0.047612513	0	100%	0.0422	0.0228	0
Configuration 19	0.0067	0	---	100%	0.067	0	---
Configuration 20	0.0033	0	---	100%	0.0033	0	---
Standard Dev	0.040504262	0.038821716	0.350869941	---	0.01800568	0.016633127	0.334556579
Average	0.083725015	0.053135807	0.150394821	---	0.044385	0.025525	0.138888889

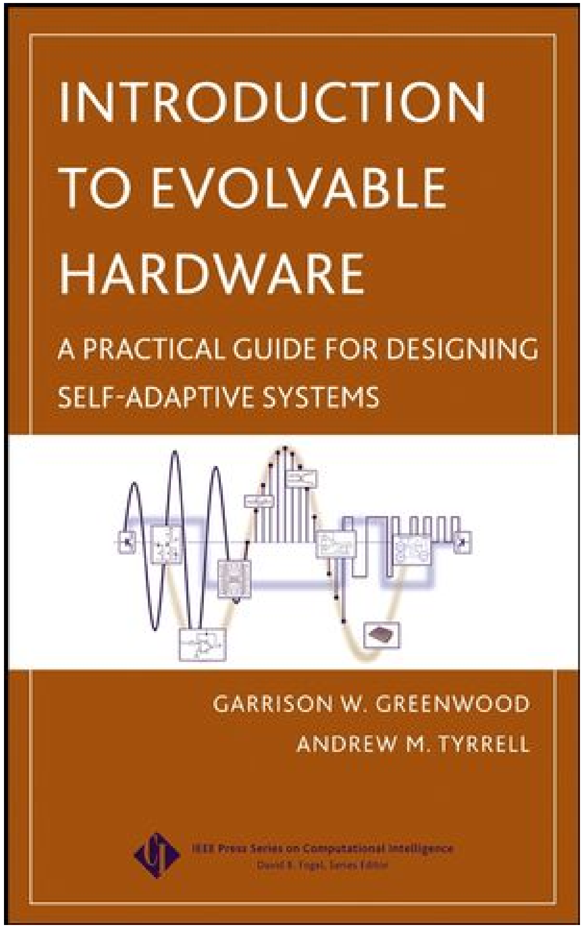
Open Issues

- Evolvable and Adaptive Hardware Applications
  - Are there practical problems that *require* the use of evolvable and adaptive hardware?
    - If those problems exists, can they be addressed with today's technology. If not, what needs to be developed?
- Evolvablity of EAH Systems
  - How does one measure the adaptability of an EAH system?
  - What adaptive learning algorithms are most likely to maximize measured adaptability?
  - Are there common features across all EAH substrates that can be isolated and exploited to improve learning and adaptation?

Open Issues

- Verification of EAH systems
  - How can one verify that an evolved system will be safe as well as effective?
  - Are there ways of explaining how evolved devices function?
  - Are there necessary trade-offs between evolvablity and understandability? If so, what are they?
- Future Directions
  - Are there new materials and/or technologies that can serve as an EAH substrate? What are they?

References: Books



Introduction to Evolvable Hardware: A Practical Guide for Designing Self-Adaptive Systems

Garrison W. Greenwood, Andrew M. Tyrrell

ISBN: 978-0-471-71977-9

Hardcover

208 pages

October 2006, Wiley-IEEE Press

References: Representative Conferences

- EAH Dedicated Events
  - Toward Evolvable Systems (1995)
  - International Conference on Evolvable Systems: From Biology to Hardware (1996 - 2007).
  - NASA/DoD {Workshop/Conference} on Evolvable Hardware (1999 - 2005)
  - NASA/ESA Conference on Adaptive Hardware and Systems (2006, 2007)
- Events Often Featuring EAH Topics and/or Tracks
  - The Genetic and Evolutionary Computation Conference (GECCO)
  - The IEEE Congress on Evolutionary Computation
  - Parallel Problem Solving from Nature (PPSN)

References: WWW References

http://carl.cs.wright.edu/eh_references.html

IEEE Computational Intelligence Society