Biologically-Inspired Massively-Parallel Architectures
- computing beyond a million processors

Dave Lester
The University of Manchester
d.lester@manchester.ac.uk
Outline

• 60 years of progress
• Computer Architecture Perspective
• Building Brains
• The SpiNNaker system
• A generic neural modelling platform
• Conclusions
ARM9 (2005)
60 years of progress

- **Baby:**
  - filled a medium-sized room
  - used 3.5 kW of electrical power
  - executed 700 instructions per second

- **ARM968:**
  - fills 0.4mm² of silicon (130nm)
  - uses 20 mW of electrical power
  - executes 200,000,000 instructions per second
Energy efficiency

• Baby:
  – 5 Joules per instruction

• ARM968:
  – 0.000 000 000 1 Joules per instruction

50,000,000,000 times better than Baby!

(James Prescott Joule born Salford, 1818)
Moore’s Law

Transistors per Intel chip

Year


Millions of transistors per chip

0.001 0.01 0.1 1 10 100

4004 8008 8080 8086 286 386 486

Pentium 4  Pentium III  Pentium  Pentium II  Pentium 8086

8080 8086 286 386 486


NeuroML March 2011
Cost of a Transistor
Moore’s Law

- SanDisk 12GB microSD
  - 50 billion transistors
  - for £20!
...the Bad News

- atomic scales
- less predictable
- less reliable
Cost of design

58% per year compound complexity growth

21% per year compound productivity growth

Logic transistors per chip

Productivity (transistors/staff month)
Outline

- 60 years of progress
- Computer Architecture Perspective
- Building Brains
  - The *SpiNNaker* system
- A generic neural modelling platform
- Conclusions
Multi-core CPUs

• High-end uniprocessors
  – diminishing returns from complexity
  – wire vs transistor delays

• Multi-core processors
  – cut-and-paste
  – *simple* way to deliver more MIPS

• Moore’s Law
  – more transistors
  – more cores

... but what about the software?
Multi-core CPUS

• General-purpose parallelization
  – an unsolved problem
  – the ‘Holy Grail’ of computer science for half a century?
  – but imperative in the many-core world

• Once solved
  – few complex cores, or many simple cores?
  – simple cores win hands-down on power-efficiency!
Back to the future

• Imagine...
  – a limitless supply of (free) processors
  – load-balancing is irrelevant
  – all that matters is:
    • the energy used to perform a computation
    • formulating the problem to avoid synchronisation
    • abandoning determinism

• How might such systems work?
Bio-inspiration

• How can massively parallel computing resources accelerate our understanding of brain function?
• How can our growing understanding of brain function point the way to more efficient parallel, fault-tolerant computation?
Outline

• 60 years of progress
• Computer Architecture Perspective
• Building Brains
  • The *SpiNNaker* system
• A generic neural modelling platform
• Conclusions
Brains demonstrate
  - massive parallelism ($10^{11}$ neurons)
  - massive connectivity ($10^{15}$ synapses)
  - excellent power-efficiency
    - much better than today’s microchips
  - low-performance components (~ 100 Hz)
  - low-speed communication (~ metres/sec)
  - adaptivity – tolerant of component failure
  - autonomous learning
Building brains

• Neurons
  • multiple inputs, single output (c.f. logic gate)
  • useful across multiple scales \((10^2 \text{ to } 10^{11})\)

• Brain structure
  • regularity
  • e.g. 6-layer cortical ‘microarchitecture’
To compute we need:
- **Processing**
- **Communication**
- **Storage**

**Processing:**
abstract model
- linear sum of weighted inputs
  - ignores non-linear processes in dendrites
- non-linear output function
- learn by adjusting synaptic weights
• Leaky integrate-and-fire model
  - inputs are a series of spikes
  - total input is a weighted sum of the spikes
  - neuron activation is the input with a “leaky” decay
  - when activation exceeds threshold, output fires
  - habituation, refractory period, ...?

\[
x_i = \sum_k \delta(t - t_{ik})
\]

\[
I = \sum_i w_i x_i
\]

\[
\dot{A} = -\frac{A}{\tau_A} + I
\]

if \( A > \vartheta_A \) fire

\& set \( A = 0 \)
• Izhikevich model
  – two variables, one fast, one slow:
    \[ \dot{v} = 0.04v^2 + 5v + 140 - u + I \]
    \[ \dot{u} = a \cdot (bv - u) \]
  – neuron fires when
    \[ V > 30 \]; then:
    \[ v = c \]
    \[ u = u + d \]
  – a, b, c & d select behaviour

(www.izhikevich.com)
Communication

• Spikes
  – biological neurons communicate principally via ‘spike’ events
  – asynchronous
  – information is only:
    • which neuron fires, and
    • when it fires
  – ‘Address Event’ Representation (AER)
Storage

• Synaptic weights
  – stable over long periods of time
    • with diverse decay properties?
  – adaptive, with diverse rules
    • Hebbian, anti-Hebbian, LTP, LTD, ...

• Axon ‘delay lines’

• Neuron dynamics
  – multiple time constants

• Dynamic network states
Outline

• 60 years of progress
• Computer Architecture Perspective
• Building Brains
• The *SpiNNaker* system
• A generic neural modelling platform
• Conclusions
SpiNNaker project

- Multi-core CPU node
  - 18 ARM968 processors
  - to model large-scale systems of spiking neurons
- Scalable up to systems with 10,000s of nodes
  - over a million processors
  - $>10^8$ MIPS total
- Power $\sim 25\mu\text{w/neuron}$
Design principles

- Virtualised topology
  - physical and logical connectivity are decoupled
- Bounded asynchrony
  - time models itself
- Energy frugality
  - processors are free
  - the real cost of computation is energy
CMP node
SpiNNaker chips
SpiNNaker test chip
Outline

• 60 years of progress
• Computer Architecture Perspective
• Building Brains
• The SpiNNaker system
• A generic neural modelling platform
• Conclusions
Event-driven software model

- Spike packet arrived
  - initiate DMA
- DMA of synaptic data completed
  - process inputs
  - insert axonal delay
- 1ms timer interrupt
  - differential equation solver
Function pipeline

SDRAM DMA Transfers

ARM968 Data-Processing Instructions

DTCM

ARM968 Subroutines

Timer (Events)

Variable Retrieval

Polynomial Evaluation

Look-Up Table

Interpolation

Differential Equation Solver
The doughnut hunter
500 neuron test

Raster plot in Matlab (fixed-point)

Raster Plot from the Test Chip

States of Neuron ID 0 on the Test Chip

- electrical current
- membrane potential
**PyNN integration**

**pyNN.SpiNNaker module**
- builds the network
- extracts the information regarding the network structure and parameters
- scripts the execution of low-level tools
- drives Ethernet interface to load, execute and retrieve results

**Initload**
- compiles the network, mapping into SpiNNaker binary data structures
PyNN integration

- LIF

- Izhikevich
• Vogels-Abbott benchmark
  – 500 LIF neurons
Telluride workshop
Robot control network

### POLARITY RETINOTOPIC MAP

16x16x2

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
</tr>
<tr>
<td>32</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>48</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>64</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
</tr>
<tr>
<td>80</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
<td>61</td>
<td>62</td>
</tr>
<tr>
<td>96</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
</tr>
<tr>
<td>112</td>
<td>80</td>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
<td>90</td>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
</tr>
<tr>
<td>128</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
<td>101</td>
<td>102</td>
<td>103</td>
<td>104</td>
<td>105</td>
<td>106</td>
<td>107</td>
<td>108</td>
<td>109</td>
<td>110</td>
</tr>
<tr>
<td>144</td>
<td>112</td>
<td>113</td>
<td>114</td>
<td>115</td>
<td>116</td>
<td>117</td>
<td>118</td>
<td>119</td>
<td>120</td>
<td>121</td>
<td>122</td>
<td>123</td>
<td>124</td>
<td>125</td>
<td>126</td>
</tr>
<tr>
<td>160</td>
<td>128</td>
<td>129</td>
<td>130</td>
<td>131</td>
<td>132</td>
<td>133</td>
<td>134</td>
<td>135</td>
<td>136</td>
<td>137</td>
<td>138</td>
<td>139</td>
<td>140</td>
<td>141</td>
<td>142</td>
</tr>
<tr>
<td>176</td>
<td>144</td>
<td>145</td>
<td>146</td>
<td>147</td>
<td>148</td>
<td>149</td>
<td>150</td>
<td>151</td>
<td>152</td>
<td>153</td>
<td>154</td>
<td>155</td>
<td>156</td>
<td>157</td>
<td>158</td>
</tr>
<tr>
<td>192</td>
<td>160</td>
<td>161</td>
<td>162</td>
<td>163</td>
<td>164</td>
<td>165</td>
<td>166</td>
<td>167</td>
<td>168</td>
<td>169</td>
<td>170</td>
<td>171</td>
<td>172</td>
<td>173</td>
<td>174</td>
</tr>
<tr>
<td>208</td>
<td>176</td>
<td>177</td>
<td>178</td>
<td>179</td>
<td>180</td>
<td>181</td>
<td>182</td>
<td>183</td>
<td>184</td>
<td>185</td>
<td>186</td>
<td>187</td>
<td>188</td>
<td>189</td>
<td>190</td>
</tr>
<tr>
<td>224</td>
<td>192</td>
<td>193</td>
<td>194</td>
<td>195</td>
<td>196</td>
<td>197</td>
<td>198</td>
<td>199</td>
<td>200</td>
<td>201</td>
<td>202</td>
<td>203</td>
<td>204</td>
<td>205</td>
<td>206</td>
</tr>
<tr>
<td>240</td>
<td>208</td>
<td>209</td>
<td>210</td>
<td>211</td>
<td>212</td>
<td>213</td>
<td>214</td>
<td>215</td>
<td>216</td>
<td>217</td>
<td>218</td>
<td>219</td>
<td>220</td>
<td>221</td>
<td>222</td>
</tr>
<tr>
<td>256</td>
<td>224</td>
<td>225</td>
<td>226</td>
<td>227</td>
<td>228</td>
<td>229</td>
<td>230</td>
<td>231</td>
<td>232</td>
<td>233</td>
<td>234</td>
<td>235</td>
<td>236</td>
<td>237</td>
<td>238</td>
</tr>
<tr>
<td>272</td>
<td>240</td>
<td>241</td>
<td>242</td>
<td>243</td>
<td>244</td>
<td>245</td>
<td>246</td>
<td>247</td>
<td>248</td>
<td>249</td>
<td>250</td>
<td>251</td>
<td>252</td>
<td>253</td>
<td>254</td>
</tr>
</tbody>
</table>

### SUBSAMPLER

4x4

(1-Winner-Take-All)

<table>
<thead>
<tr>
<th>L</th>
<th>F</th>
<th>F</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>F</td>
<td>F</td>
<td>R</td>
</tr>
<tr>
<td>L</td>
<td>F</td>
<td>F</td>
<td>R</td>
</tr>
<tr>
<td>L</td>
<td>F</td>
<td>F</td>
<td>R</td>
</tr>
</tbody>
</table>

### OUTPUTS

(1-Winner-Take-All)
Robot control network
Outline

• 60 years of progress
• Computer Architecture Perspective
• Building Brains
• The SpiNNaker system
• A generic neural modelling platform
• Conclusions
Current status...

• Test chip: 2 ARM968 processors
• Test card: 4 test chips, 8 processors
  – Cards can be linked together
• Neuron models: LIF, Izhikevich, MLP
• Synapse models: STDP, NMDA
• Networks: PyNN -> SpiNNaker, various small tools to build Router tables, etc
• Monitor/debug: SpiNN doctor
...and the next steps

• Full (18-processor) chip: tape-out end Oct’10
  – Package silicon with SDRAM in February 2011
  – Build 4-chip test card (Feb’11),
    50-chip 103 machine (Q2 2011), 500-chip 104 machine (H2 2011), 5,000-chip 105 machine (H1 2012), 50,000-chip 106 machine (H2 2012).
  – Rebuild event-driven software foundations
  – Extend PyNN -> SpiNNaker support
  – Monitor/debug tools,
    developmental models, intrinsic configuration, run-time fault-tolerance,...
Conclusions

- Brains represent a significant computational challenge
  - now coming within range?
- **SpiNNaker** is driven by the brain modelling objective
  - virtualised topology, bounded asynchrony, energy frugality
- The major architectural innovation is the multicast communications infrastructure
- Fault-tolerance has been considered throughout
  - though the approach is rather ‘ad hoc’
- We have prototype working hardware!
SpiNNaker team

Manchester

Southampton