Computerschach

Die letzte Runde im Mensch-Maschine Schachwettkampf ist beendet

Anmerkung: Dieses Kapitel enthält Aussagen, die man zu Recht als „nicht-wissenschaftlich“ bezeichnen sollte. Es handelt sich dabei um reine Spekulation, und sie wurden deshalb als solche gekennzeichnet. Dies gilt insbesondere für die Folien 109 und 110.

Computerschach hat eine atemberaubende Geschichte

TECHNISCHE UNIVERSITÄT DARMSTADT

1940-1970: Versuche, menschliches Schachspiel nachzuahmen, versanden

1. 1970s: Chess 4.5 ist das erste 'starke' Programm. Es legt Wert auf Sucheffizienz. Erster Computersieg eines Menschenturniers, Minnesota Open 1977.
2. 1983: Belle wurde 'National Master', 2100 ELO.
3. 1988: Hitec erringt einen ersten Sieg gegen einen Großmeister
4. 1988: Deep Thought auf Großmeisterniveau
5. 1992: Die ChessMachine gewinnt die offene Computerschachweltmeisterschaft, ein konventionelles PC-Programm von Ed Schröder.
6. 1997: Deep Blue schlägt Kasparov in einem 6-Spiele Wettkampf.
7. Von der Zeit an, dominieren PC-Programme die Welt, mit einer jährliche Spielstärke-steigerung von ca. 30 ELO. Im Internet gibt es virtuelle Räume, in denen Turniere abgehalten werde, in denen man jederzeit gegen Großmeister oder starke Maschinen spielen kann.

ELO: statistisches Maß; 100 Pkte Differenz entsprechen einer 64% Gewinnchance

Anfänger 1000ELO
Internationaler Meister > 2400
Großmeister > 2500
Menschlicher Weltmeister > 2800

Das Ziel

Nur ein BIG POINT fehlte noch:
stärker spielen als die besten Menschen.

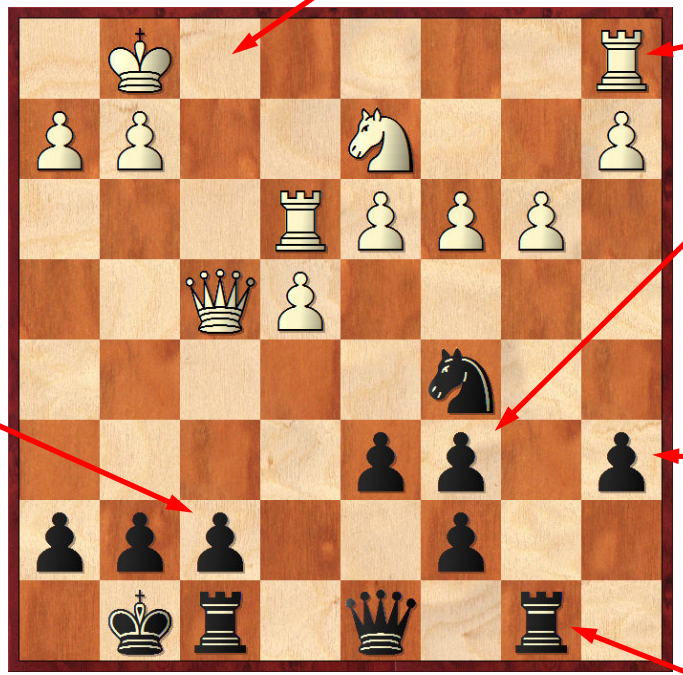
4 Programme lieferten sich bis 2005 ein Rennen:

- **Shredder** von Stefan Meyer-Kahlen,
- **Fritz** von Frans Morsch,
- **Junior** von Amir Ban und Shay Bushinsky
- **Brutus/Hydra** von Chrilly Donninger, Ulf Lorenz (2003-2006), Christopher Lutz, Nasir Ali
- Rybka, Fruit seit 2005

Die oberen vier Programme holten z.B. auf der Computerschach-Weltmeisterschaft in Graz, 2003, mehr als 95% der Punkte gegen das restliche Feld.

Was macht ein Schachprogramm eigentlich?

(I) Stellungsbewertung



London: Die Kontrahenten

TECHNISCHE UNIVERSITÄT DARMSTADT

Adams:

geboren 17.11.1971, lebt in London.

- Wurde 1989 Großmeister, mit dem Gewinn der Britischen Meisterschaften, im Alter von 17 Jahren.
- 1997 britischer Meister.
- 1990 und 2005 zum Spieler des Jahres gekürt und
- gewann zwischen 1993 und 2002 rekordverdächtig viele Turniere.

Zu der Zeit Nr. 1 in England, Nr. 7 der Welt, mit einer Spielstärke von 2741 Elo.

London: Die Kontrahenten

TECHNISCHE UNIVERSITÄT DARMSTADT

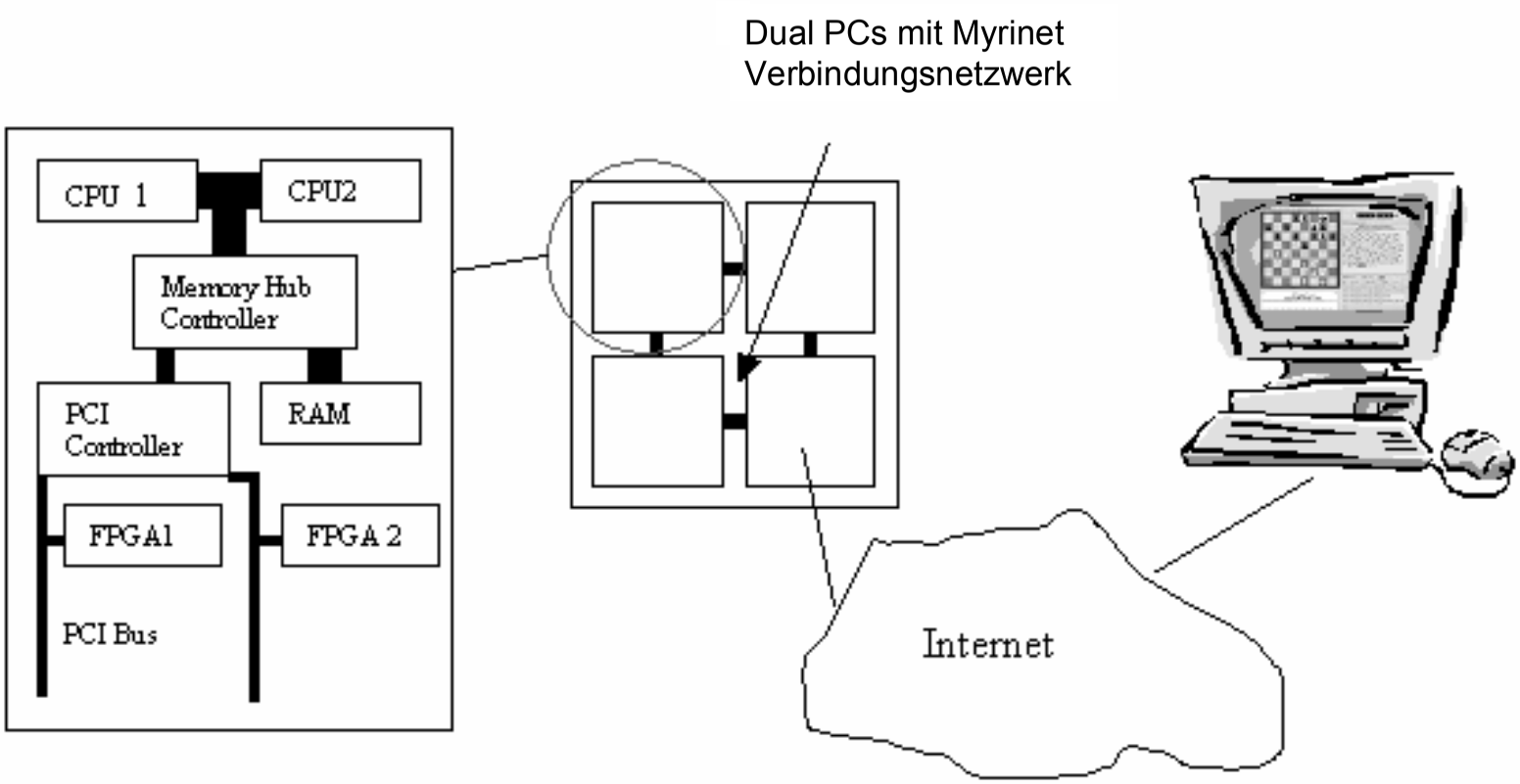
Hydra:

- Die einzige 'Schacheinheit', die jemals (bis 2009) deutlich über 3000 Elo spielt.
- 150 Millionen Schachstellungen pro Sekunde
- typische Rechentiefe nach der Eröffnung: 19 bis 20 Halbzüge
- Cluster mit 32 Pentium Prozessoren, jeweils mit einem XilinX Virtex Pro II 7000 als Schach-Koprozessor

(finanziert und durchgeführt von PAL Computer Systems, VAE, Abu Dhabi)

- Gewinner des 13. IPCCC 2004 mit 6.5 aus 7
- Gewinner gegen Shredder in Abu Dhabi mit 5.5 : 2.5
- Gewinner gegen GM Vladimirov (2630 Elo) mit 3.5 : 0.5
- Mensch-Maschine Mannschaftsweltmeister mit 3.5 : 0.5 gegen Schnitt von 2690 ELO
- Gewinner des 14. IPCCC 2005 mit 8 aus 9

Hydras globale Architektur

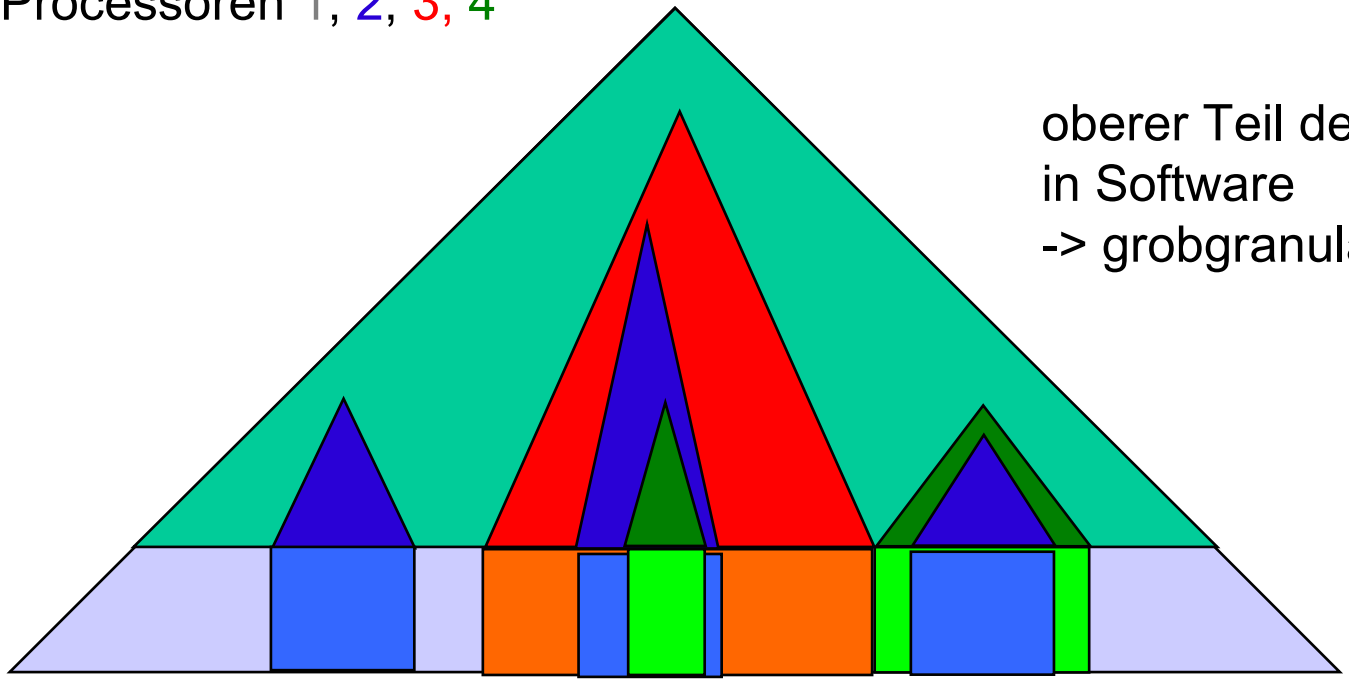


FPGA: Virtex II: VP70 -> ca. 5,4 Millionen Suchknoten pro Sekunde pro Prozessor
bis zu 32 CPUs: 3.0 GHz -> Speedup 17 und somit 100 Millionen Suchknoten pro Sekunde

Zerlegung des Suchbaums

TECHNISCHE UNIVERSITÄT DARMSTADT

Processoren 1, 2, 3, 4



oberer Teil des Baums
in Software
-> grobgranulare Parallelität

die letzten 4 Ebenen in FPGA -> feingranulare Parallelität,
FPGA Karte wird ca. 100000 mal pro Sekunde angestoßen

Paralleler Suchalgorithmus

TECHNISCHE UNIVERSITÄT DARMSTADT

- Zerlegung des Suchbaums
- ‘Work Stealing‘: Prozessoren ohne Arbeit senden randomisiert Anfragen
  in den Cluster;
  ein spezieller Prozessor beginnt die Berechnungen wie im sequentiellen Fall
- ein Prozessor, der Arbeit hat und eine Arbeitsanfrage einfängt, gibt ein Teilproblem
  an den Sender der Anfrage ab,
  der Anfragende wird 'worker‘, der andere ‘master‘
- master/worker Beziehungen sind dynamisch, oft geknüpft und wieder gelöst

- Nachrichten: REQUEST, WORK, NO-WORK, WINDOW_UPDATE,
  CUTOFF, RESULT

- Problemauswahl für Abgaben: nah an der Wurzel, YBWC

Speedups

	Zeit(s)	SPE	SO %
1	24213	1	0
2	12139	1.99	0
4	6888	3.5	3.6
8	3488	6.5	-1
16	2011	12	9.3
32	1424 *	17	80 *

* geschätzt, da andere Versionen von Hydra zugrunde liegen

FPGA (Field Programmable Gate Array)

TECHNISCHE UNIVERSITÄT DARMSTADT

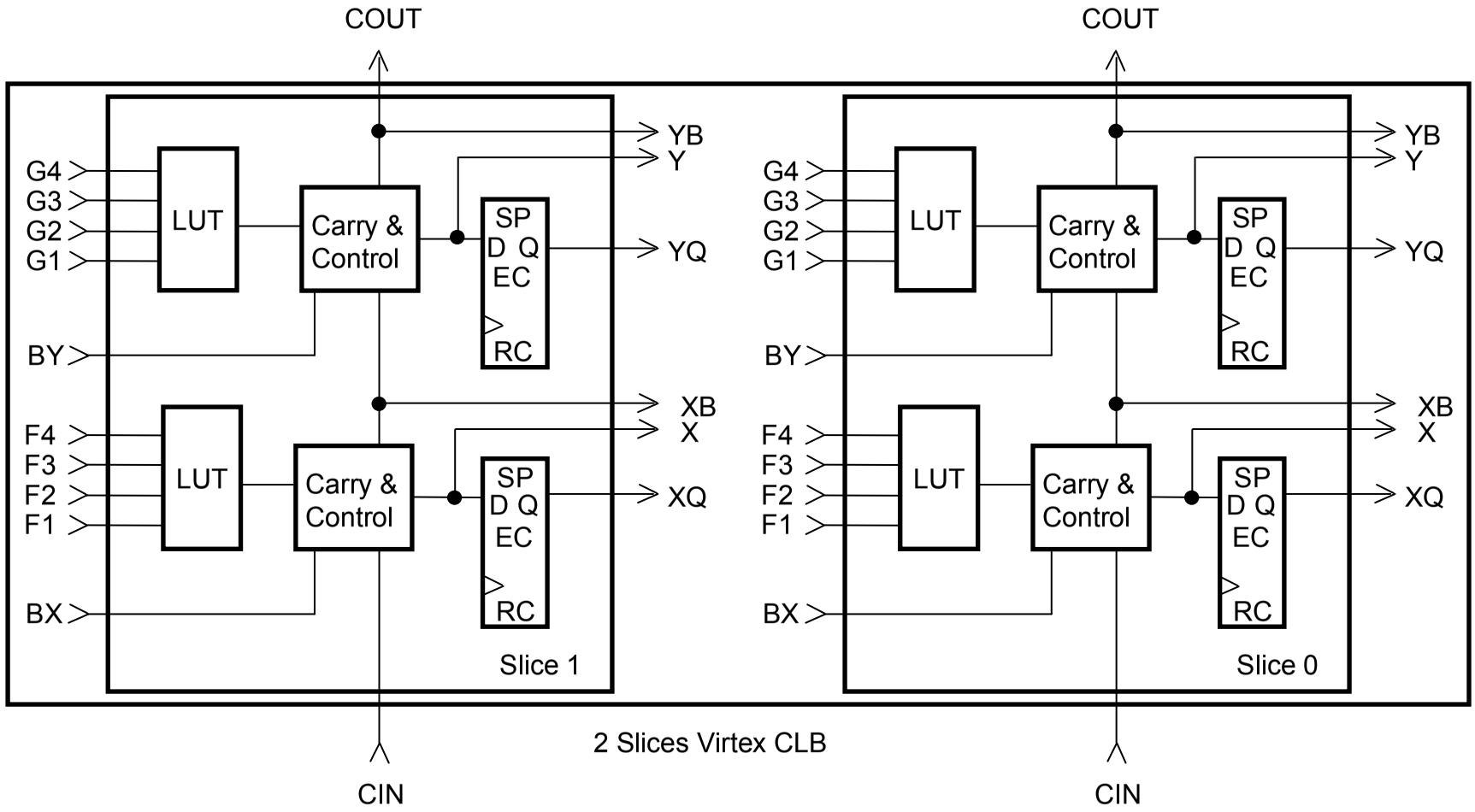
- **ist rekonfigurierbare Logic für rapid prototyping und für die Implementierung digitaler Systeme**
- **ist eine Art in RAM simulierter Hardware, die fein-granulare Parallelität erhaltend**
- **ist eine Möglichkeit Logik zu realisieren**

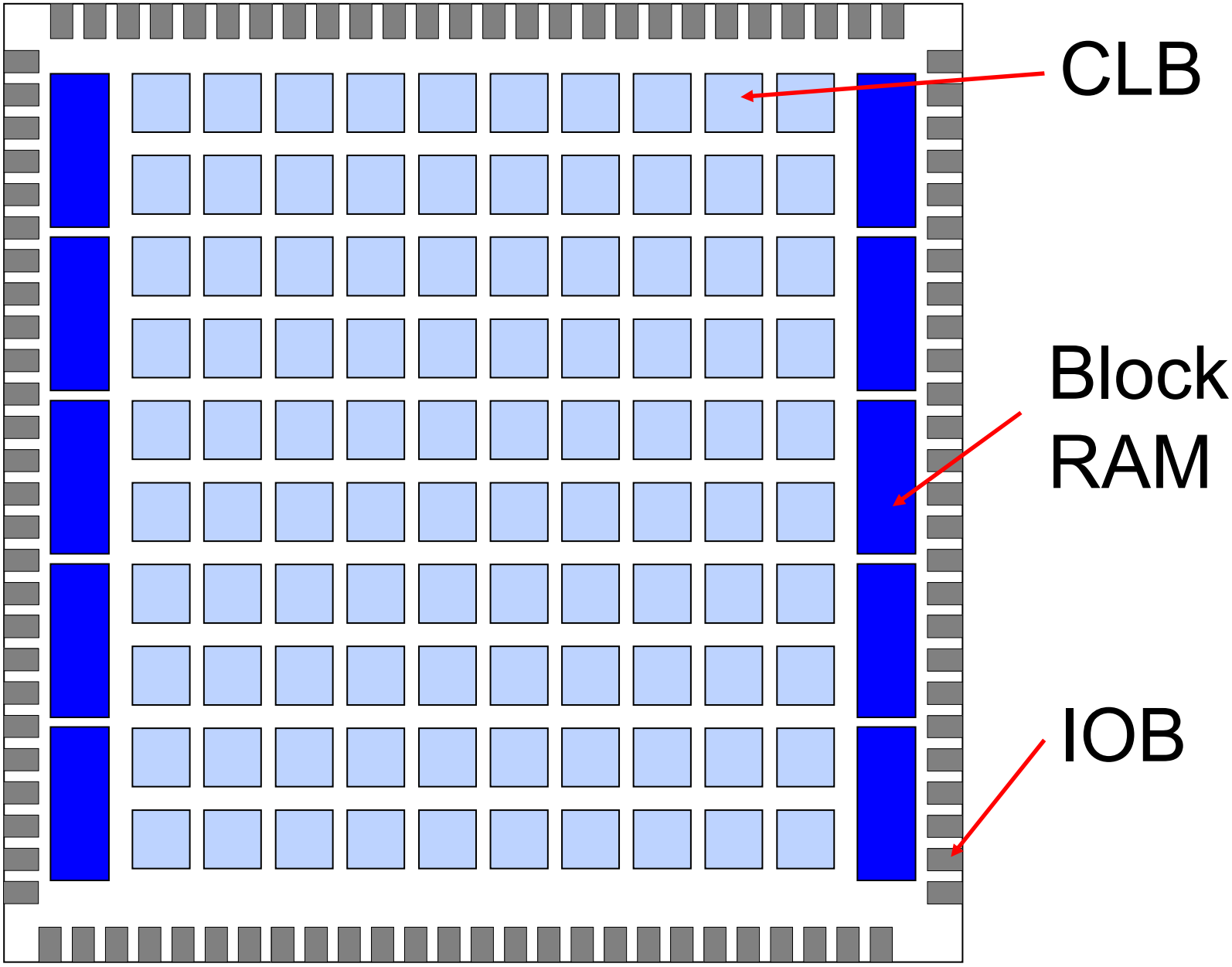
Hauptvorteile (für Hydra):

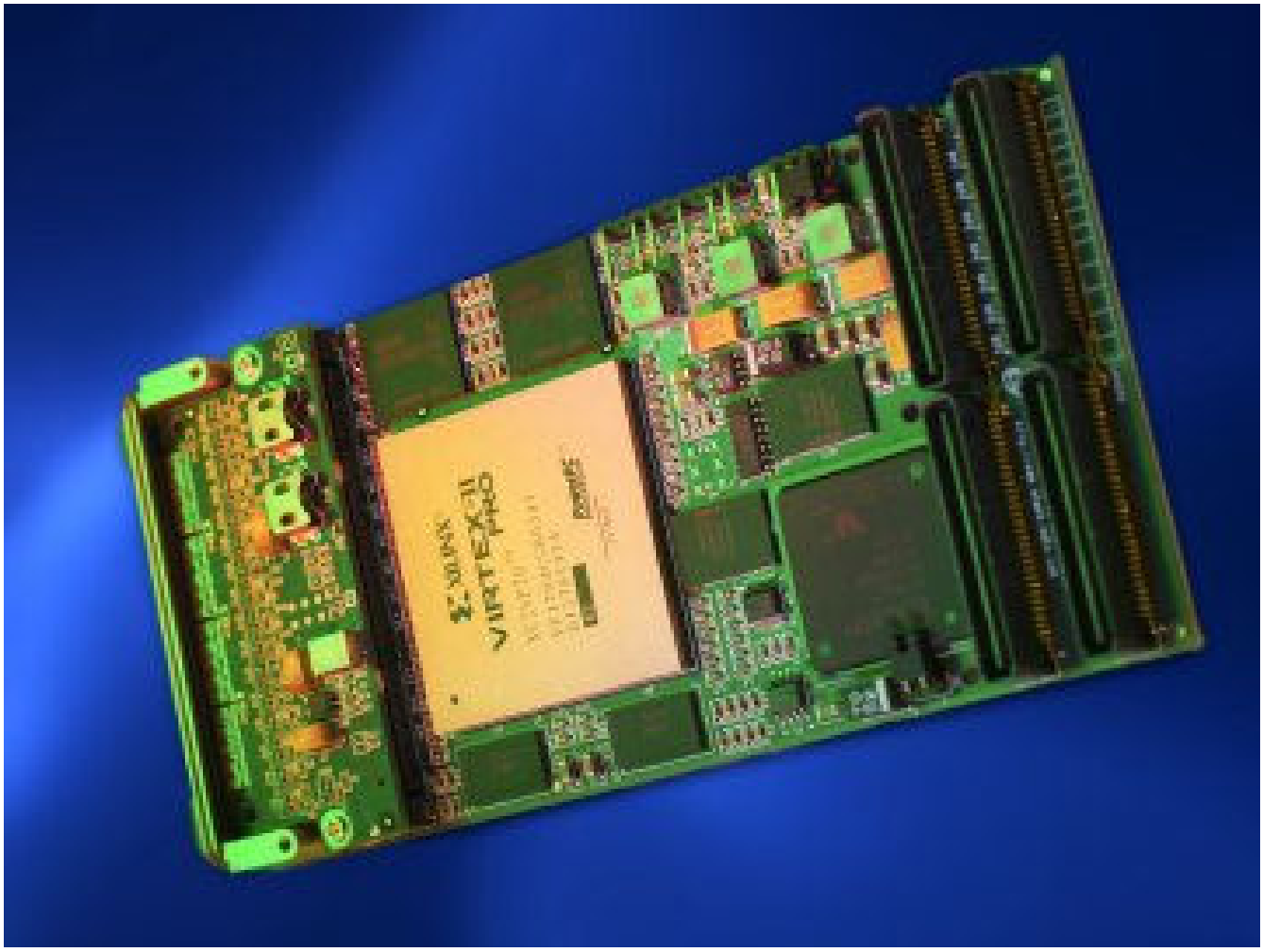
1. Das Hinzufügen von mehr Schachwissen erfordert zusätzlichen Platz, kostet aber nahezu keine zusätzliche Berechnungszeit

2. FPGA Code kann ge-debugged und schnell wie in Software geändert werden, ohne langwierige ASIC Entwicklungszeiten

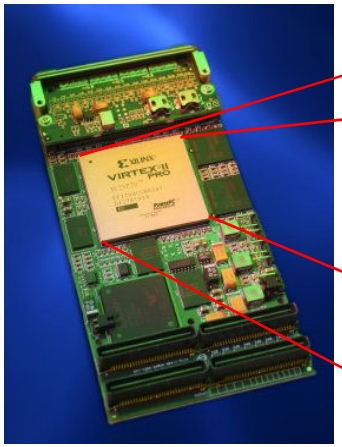
3. feingranulare Parallelität kann genutzt werden -> genMove, doMove, eval, undoMove in 9 Taktzyklen, bei einer clock-rate von 50MHz

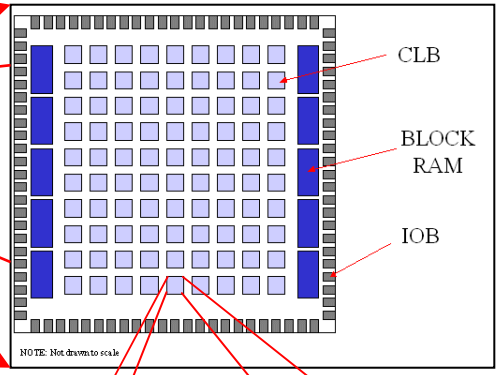

COUT
COUT
YB
Y
YQ
XB
X
XQ
G4
G3
G2
G1
BY
F4
F3
F2
F1
BX
LUT
Carry &
Control
SP
D Q
EC
RC
Slice 1
Slice 0
2 Slices Virtex CLB
CIN
CIN

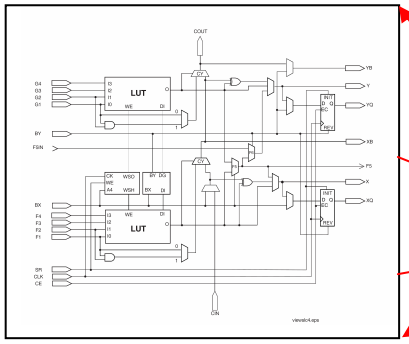

CLB
Block
RAM
IOB


VIRTEX-II
PRO

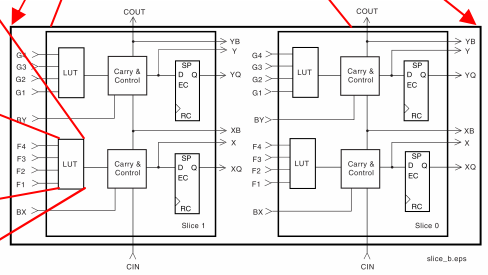
HPRCC
High Performance Reconfigurable Cluster-Computing







Detailed View of Virtex Slice

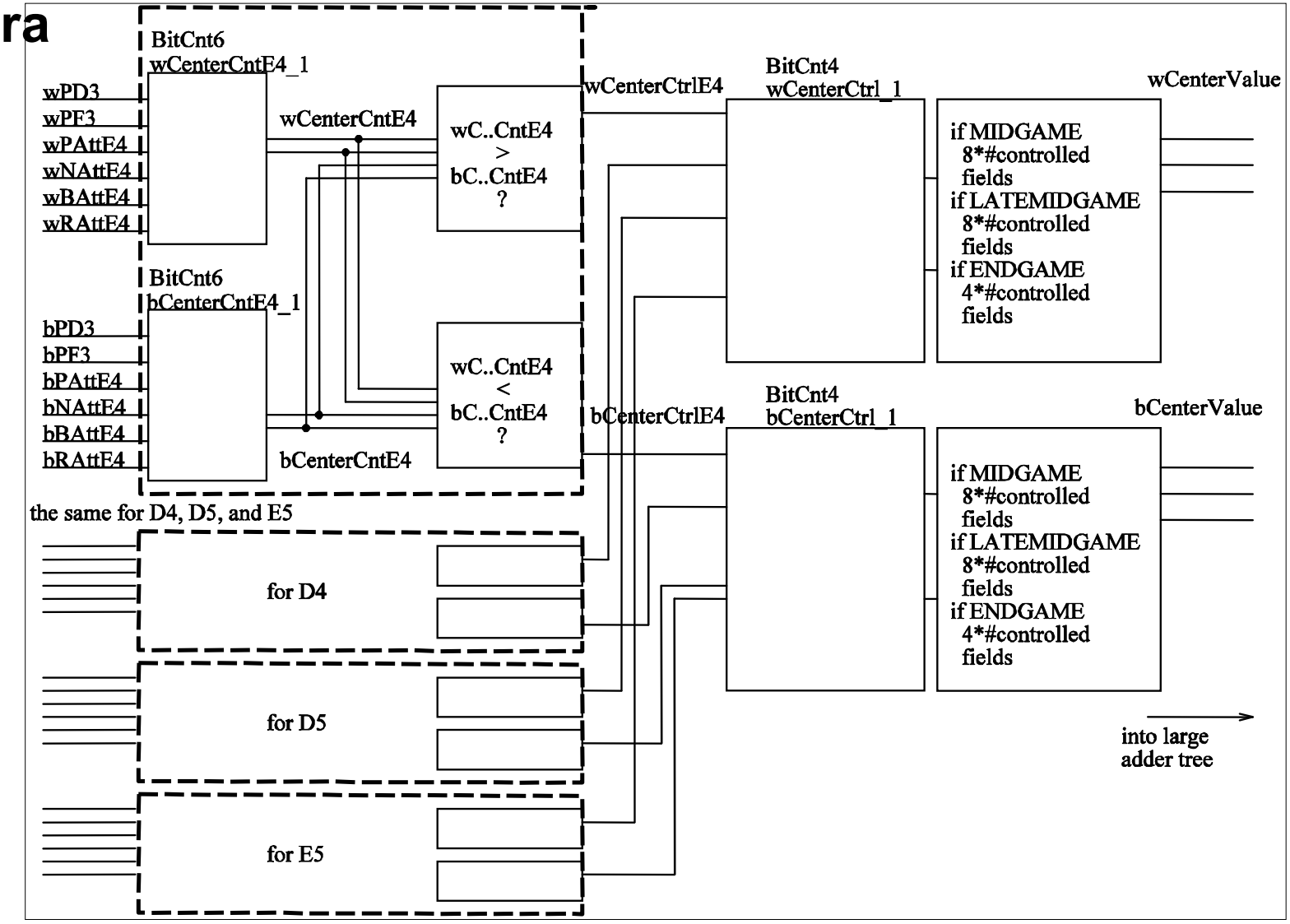


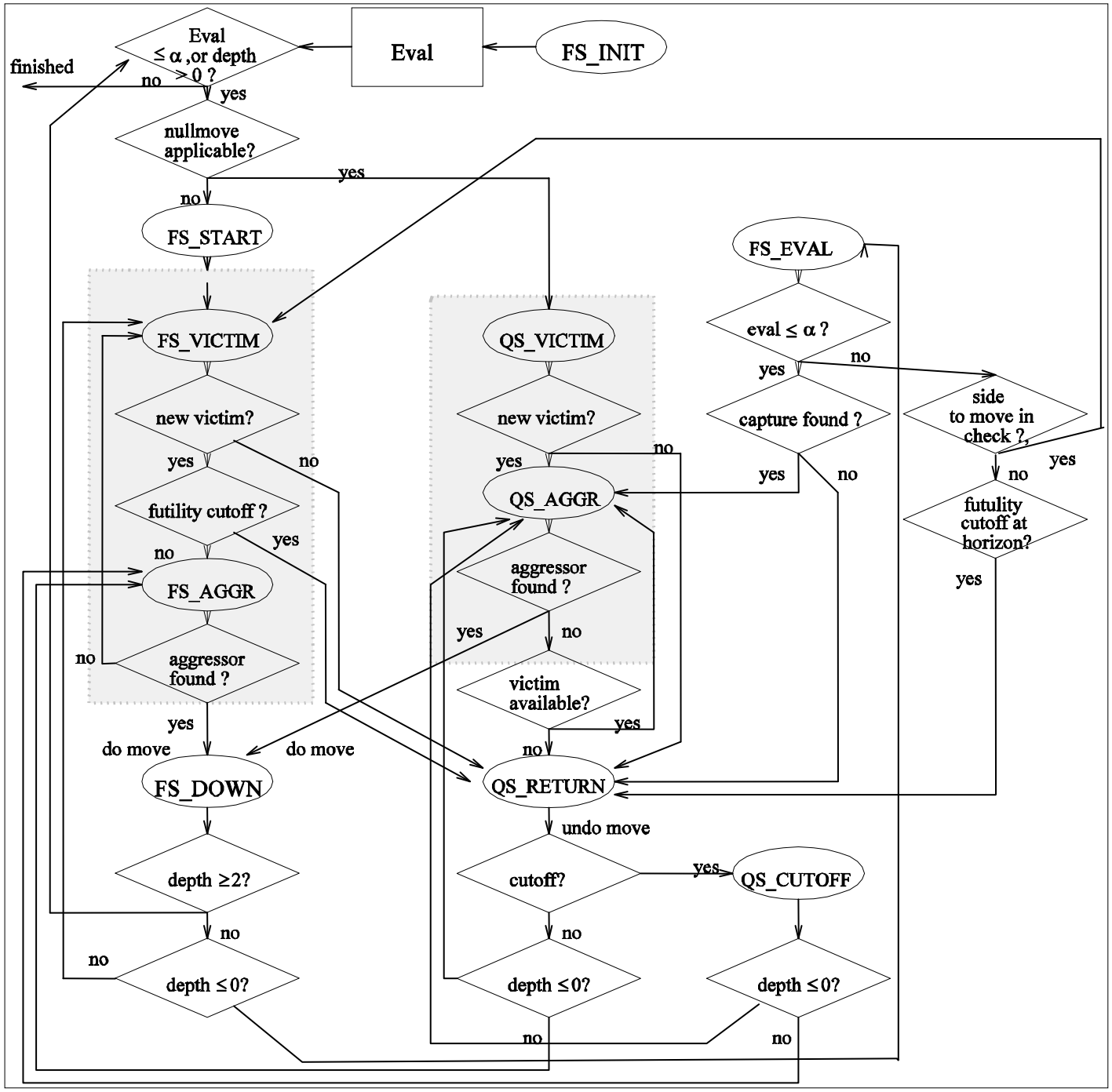
2 Slices Virtex CLB

FPGA, Verilog

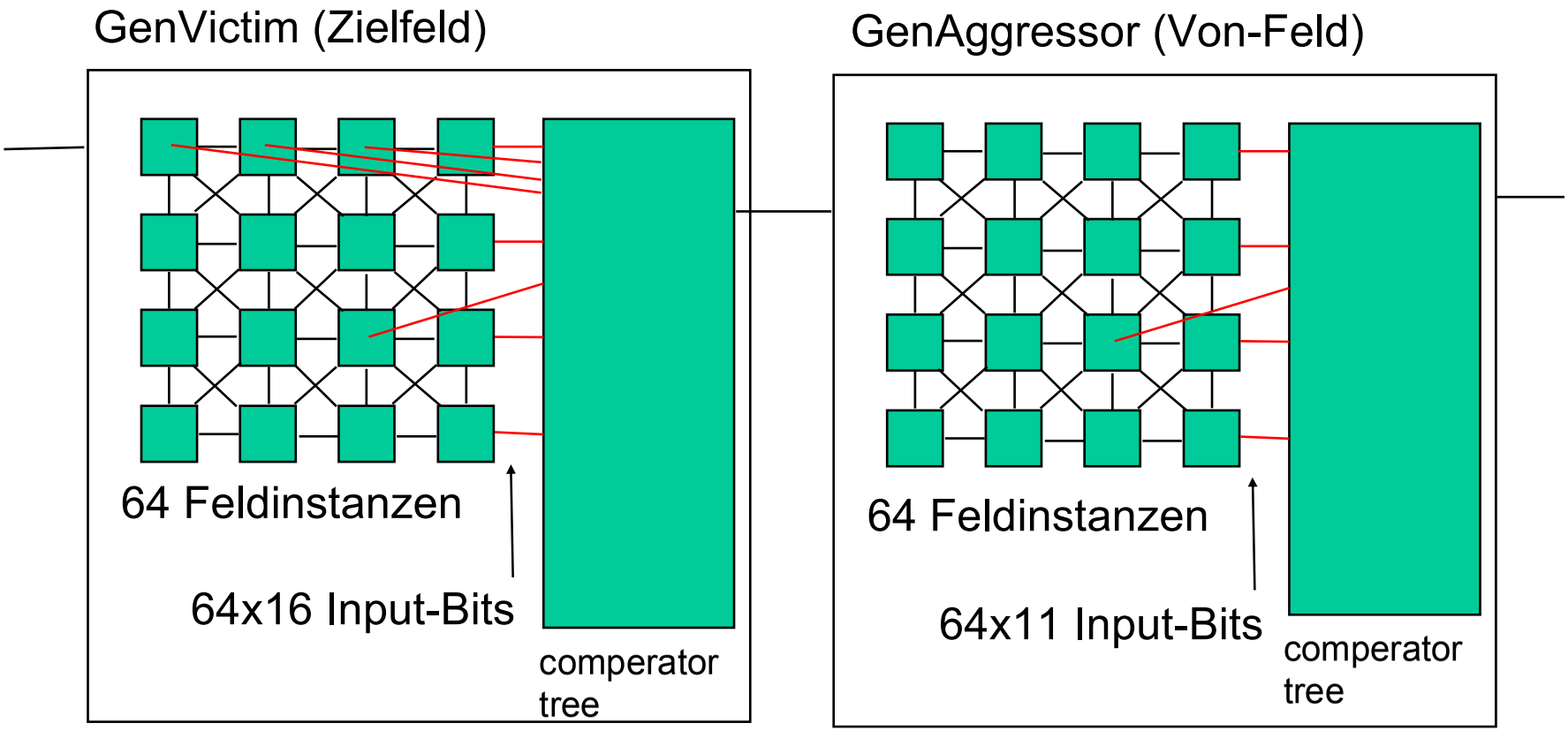
```
module Search();
  ControlLogic;
          // Enables/disables the modules and converts data to the correct format.
          // E.g. the module Board does not know whether DoMove or UndoMove is
          // done. The input is always piece-from, piece-to. The
          // ControlLogic converts the move
          // information in the correct format for the Board module.
  AlphaBetaFSM;   // Alphabeta search with Finite State Machine.
          // The FSM sets a few signals like "DoMove", "UndoMove",
          // "Next-Victim". The next state is determined from the
          // output of the modules and the current state. The signals are used by the
          // ControlLogic to control the operation of the modules.
          // The FSM has 54-States, but some states are simply waitstates. It changes no
          // signals. Only a transition to the next
          // state can be done.
  CastleStatus();      // Determine castle status                                  0.5 cycles
  GenVictim();         // Generates next ToSquare.                                 2.0 cycles
  GenAggressor();    // Generates next FromSquare for a given ToSquare.  2.0 cycles
  TestCheck();        // Tests if the king is in check.                           2.0 cycles
  Board();             // Board respresenation and board update                    1.0 cycles
  SearchStack();     // "Local variables" for the search control.                 0.5 cycles.
  Evaluate();          // Evaluation.                                              3.0 cycles.
endmodule

module Evaluate();
  HandleSpecialCases; // E.g: Is there insufficient mate material?
  WeightedSum;          // Different weights for different game phases.
  GamePhase();          // Determines the game phase.
  PieceValues();        // Sums up the material value.
  FirstOrderValues();   // piece square tables
  PawnStructure();      // pawn structure and passed pawns
  KingCover();          // pawn-cover around the king
  Dynamic();            // Main part. This evaluation bases on attack and defense
                        // relations. E.g. attack on opponents king, mobility
endmodule
```

Hydra


BitCnt6
wCenterCntE4_1
wPD3
wPF3
wPAttE4
wNAttE4
wBAttE4
wRAttE4
wCenterCntE4
wC..CntE4
>
bC..CntE4
?
wCenterCtrlE4
BitCnt4
wCenterCtrl_1
wCenterValue
if MIDGAME
8*#controlled
fields
if LATEMIDGAME
8*#controlled
fields
if ENDGAME
4*#controlled
fields
BitCnt6
bCenterCntE4_1
bPD3
bPF3
bPAttE4
bNAttE4
bBAttE4
bRAttE4
bCenterCntE4
wC..CntE4
<
bC..CntE4
?
bCenterCtrlE4
BitCnt4
bCenterCtrl_1
bCenterValue
if MIDGAME
8*#controlled
fields
if LATEMIDGAME
8*#controlled
fields
if ENDGAME
4*#controlled
fields
the same for D4, D5, and E5
for D4
for D5
for E5
into large
adder tree


Eval
≤ α ,or depth
> 0 ?
Eval
FS_INIT
finished
no
yes
nullmove
applicable?
yes
no
FS_START
FS_VICTIM
new victim?
yes
no
futility cutoff ?
yes
no
FS_AGGR
aggressor
found ?
no
yes
do move
FS_DOWN
depth ≥2?
no
no
depth ≤0?
QS_VICTIM
new victim?
yes
no
QS_AGGR
aggressor
found ?
yes
no
victim
available?
yes
no
QS_RETURN
undo move
cutoff?
yes
QS_CUTOFF
no
depth ≤0?
no
FS_EVAL
eval ≤ α ?
yes
no
capture found ?
yes
no
side
to move in
check ?,
yes
no
futulity
cutoff at
horizon?
yes
do move

Hydra



1. Besetzte Felder senden ein Signal in GenVictim.
2. Freie Felder leiten Signale weiter
3. Alle Felder, die ein Signal bekommen, sind potentielle Zielfelder.
4. Der Comperator-tree wählt attraktivstes Zielfeld (z.B. Schlagzug).

1. Gewinnerfeld generiert Signal von 'super piece'.
2. Freie Felder leiten das Signal weiter.
3. Von eigenen Figuren besetzte Felder sind potentielle Von-Felder.
4. Der Comperator-tree wählt das attraktivste Von-Feld

Das Entwicklerteam hinter Hydra

TECHNISCHE UNIVERSITÄT DARMSTADT

Dr. Chrilly Donninger

GM Christopher Lutz

Dr. Ulf Lorenz

Universität Paderborn

Prof. Dr. B. Monien

Paderborn Center for Parallel Computing

Nasir Ali

PAL Computer Systems

PAL Gruppe Abu Dhabi

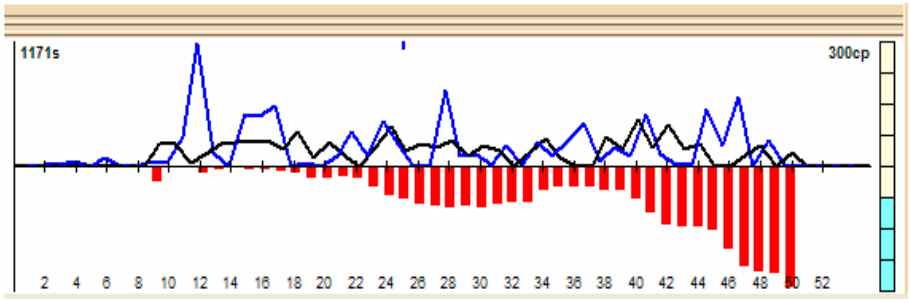
Scheich Tahnoon

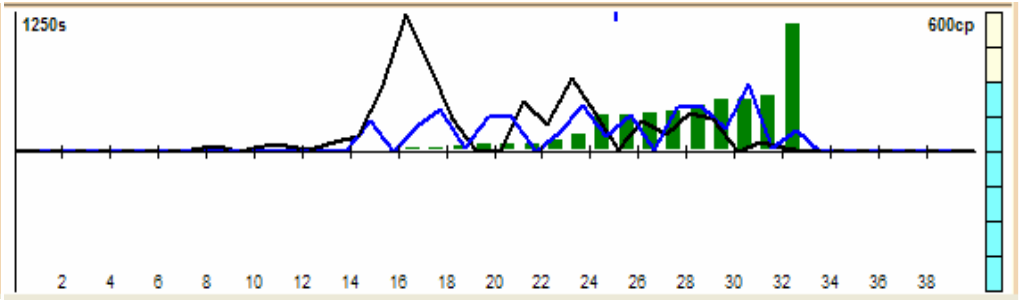
London: Hydra gegen Adams 5,5 : 0,5

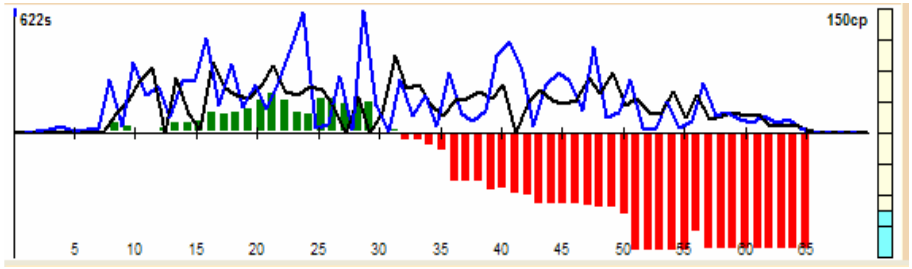
TECHNISCHE UNIVERSITÄT DARMSTADT

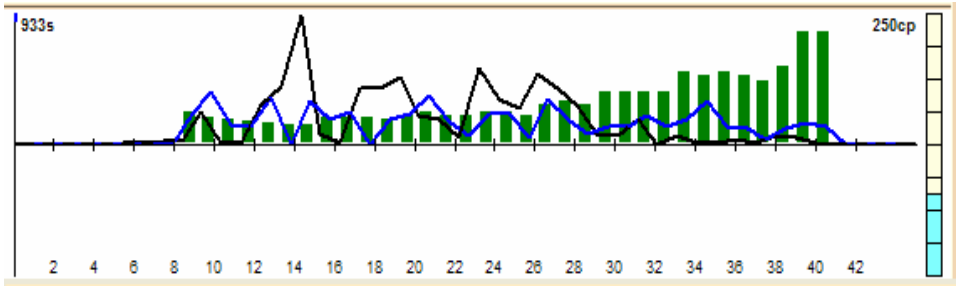
London 2005

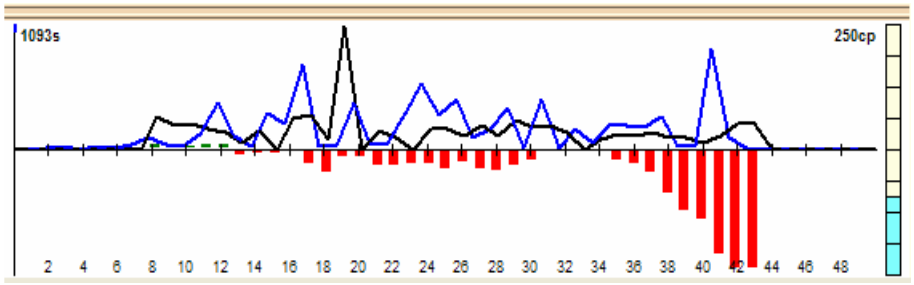
TECHNISCHE UNIVERSITÄT DARMSTADT

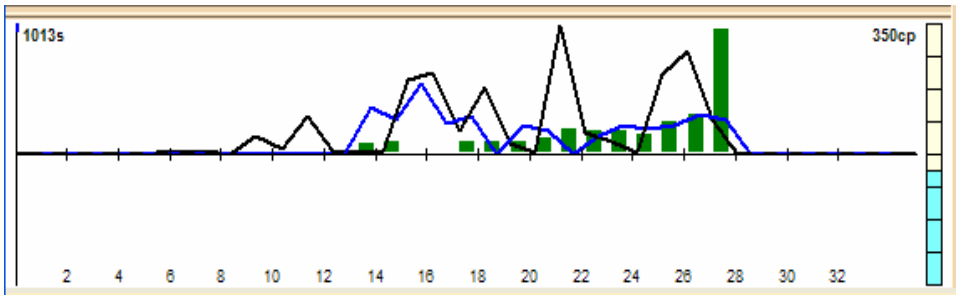












Typische Thesen aus den 80er Jahren (*) :

- Top-Großmeister spielen fast perfekt Schach
- Ein Top-Großmeister kann mit weiß immer Remis halten, wenn er nur kein Risiko eingeht und von vornherein auf Remis spielt.
- 3000 Elo ist eine natürliche Perfektionsgrenze
- Turmendspiele mit einem Wenigerbauern sind einfach Remis zu halten.

Alles Falsch!

Katastrophale Selbsteinschätzung.

(*) Diese Folie präsentiert auch aus Sicht Ihres Dozenten keine wissenschaftliche Erkenntnis. Sie dient allein der Anregung zu Diskussion!

Ein zarter Hauch von Intelligenz (*)

TECHNISCHE UNIVERSITÄT DARMSTADT

1. **Die Menschlichen Meister beanspruchen, daß sie, obwohl sie gegen ein Programm verlieren, das Spiel besser verstehen. Sie sprechen von einem 'tieferen Verständnis'.**

Frage: Tun sie das? Was könnte das sein: 'tiefer verstehen'? Gibt es überhaupt etwas zu verstehen, außer 'Jede Stellung ist entweder gewonnen, verloren oder remis'?

Antwort: Ja, es gibt da was ...

(*) Diese Folie präsentiert auch aus Sicht Ihres Dozenten keine wissenschaftliche Erkenntnis. Sie dient allein der Anregung zu Diskussion!