НИИ МВС

НИИ МВС имени академика А.В. Каляева Южного федерального университета

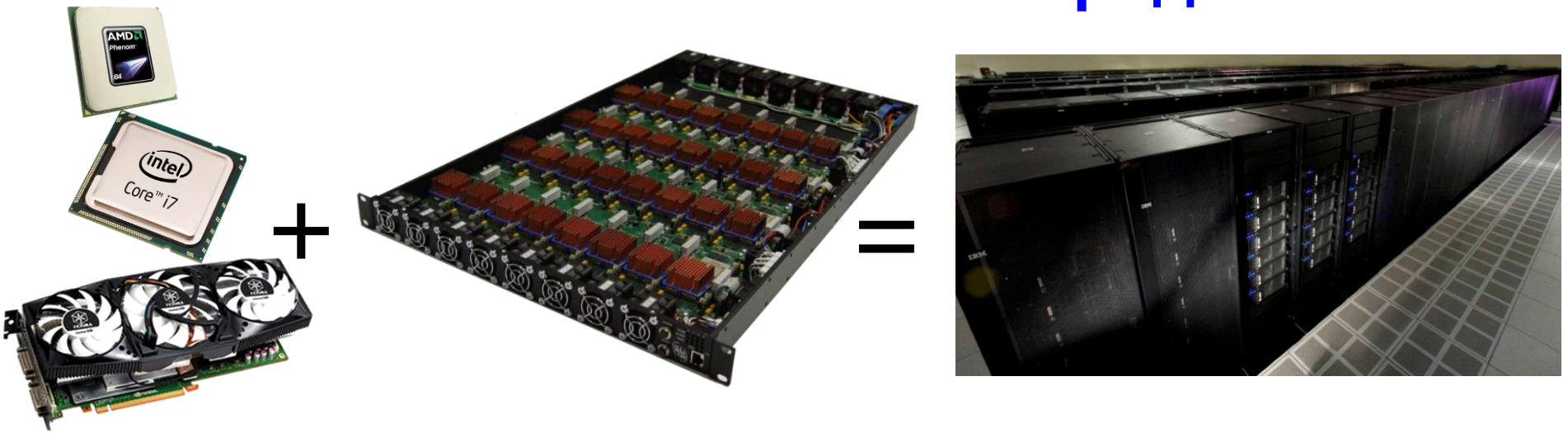
Научно-исследовательский центр супер-ЭВМ и нейрокомпьютеров

СЭ НЦ НК

Программирование вычислительных систем гибридного типа на основе метода редукции производительности

Докладчик: А.И. Дордопуло

Вычислительные системы гибридного типа



Гомогенная ВС - архитектурно-одинаковые вычислительные узлы одного типа с одинаковым типом организации вычислений и способом обработки информации

Гетерогенная ВС - архитектурно-одинаковые вычислительные узлы разных типов с одинаковым типом организации вычислений и способом обработки информации

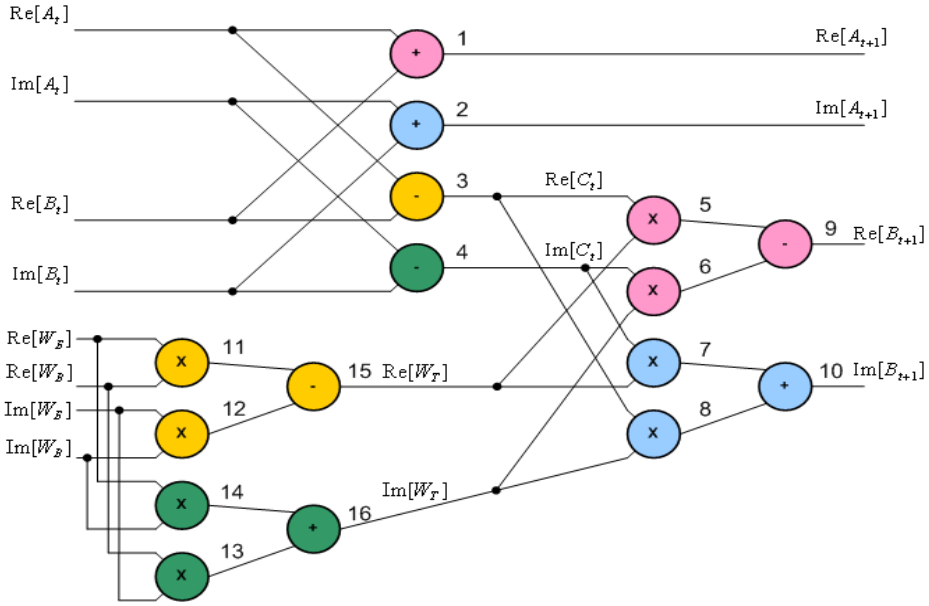
Гибридная ВС – архитектурно-различные вычислительные узлы с различным способом обработки информации и организации вычислений

Вычислительная система гибридного типа - архитектурно-различные вычислительные узлы с различным типом организации вычислений и одинаковым способом обработки информации

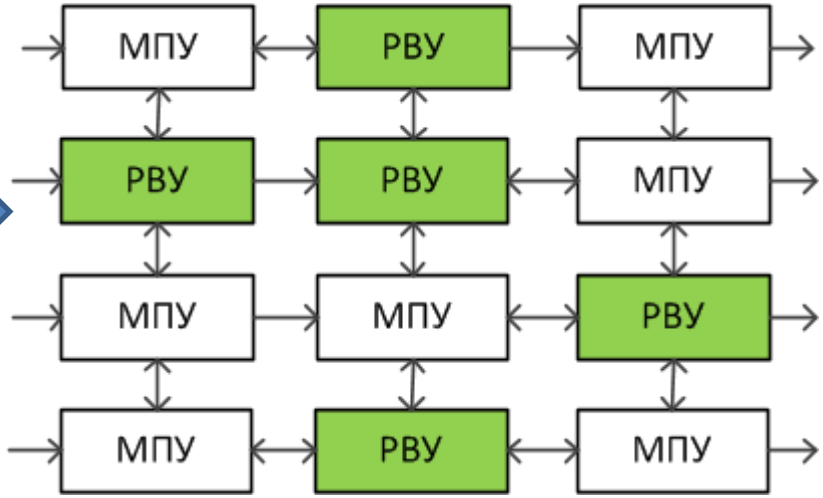
Проблема программирования ВСГТ

Создание специализированной вычислительной системы под каждую прикладную задачу является экономически нецелесообразным. Поэтому возникает проблема ресурсонезависимого программирования ВСГТ, т.е. создания прикладных программ, объединяющих в едином вычислительном контуре как последовательные, так и параллельные вычислительные фрагменты и не привязанных к конкретной архитектуре системы.

Информационный граф задачи



Архитектура ВСГТ



Существующие технологии программирования ВСГТ

1. Все технологии ориентированы на программирование только одного устройства с фиксированной внутренней структурой и заранее определенным набором команд, что усложняет масштабирование прикладной программы вычислительной системы гибридного типа, особенно в случае сокращения доступного аппаратного ресурса;

2. Каждая часть вычислительной системы гибридного типа программируется в рамках своей технологии (CUDA, OpenACC, OpenCL) отдельно на своем языке программирования.

3. Прикладная программа пишется под текущую конфигурацию вычислительной системы гибридного типа, любое изменение структуры системы приводит к изменениям программы.

4. Синхронизация информационных потоков в структуре задачи возлагается на программиста.

5. Портирование прикладной программы на другую систему с похожей конфигурацией приводит к полной переработке программы.

6. Время программирования и отладки прикладной задачи для вычислительной системы гибридного типа составляет от 6 до 12 месяцев.

Требования к технологии программирования вычислительной системы гибридного типа

1. Технология должна содержать средства описания структурных, структурно-процедурных (в т.ч. конвейерных, макроконвейерных, конвейерно- конвейерных) и мультипроцедурных вычислений и их различных сочетаний в едином вычислительном контуре.
2. Переход от одного типа организации параллельных вычислений к другому при изменении конфигурации ВСГТ должен осуществляться автоматизированно, без существенной переработки программы.
3. Прикладная задача для вычислительной системы гибридного типа должна описываться на одном языке программирования высокого уровня.
4. Необходимы средства быстрого масштабирования и изменения прикладной программы под новую конфигурацию и архитектуру вычислительной системы гибридного типа.
5. Преобразования должны осуществляться как в случае увеличения, так и в случае сокращения вычислительного ресурса.

Предлагаемое решение

1. Описание вычислений на языке программирования высокого уровня COLAMO в канонической параллельно-конвейерной форме.
2. Преобразование программы под текущую конфигурацию ВСГТ осуществляется препроцессором на основе методов редукции производительности.
3. Основным критерием получаемого решения является сбалансированность.

Преобразование программы в каноническую параллельно-конвейерную форму

1) Все массивы в параллельной программе на языке программирования высокого уровня COLAMO становятся двумерными с параллельным (Vector) и последовательный (Stream) типами доступа.

2) Все переменные параллельной программы (кроме счетчиков цикла) преобразуются в конструкцию union для непосредственного обращения к переменной по её типу, параллельного (bitvector) и последовательного (bitstream) доступа.

3) Во все циклы параллельной программы на языке программирования высокого уровня COLAMO с обработкой модернизированных массивов и переменных добавляется дополнительный цикл по добавленному измерению (по умолчанию число итераций цикла равно 1).

4) Все подкадры из параллельной программы на языке COLAMO преобразуются в конструкции Implicit с неявным типом выполнения (структурно или процедурно).

Преобразование описаний переменных

Исходный текст программы	Параллельно-конвейерная форма

Var i, j: **Number**;
 	
const N = 100;
const M = 1;
 	
Var a_s, b_s, c_s, d_s : **Array Integer** [N : **Stream**] **Mem**;
	Var i, j: **Number**;
	
	Const N = 100;
	Const M = 1;
	
	Const VEC_SIZE_0 = 1;
	
	Const STREAM_SIZE_0=N/VEC_SIZE_0;
	
	Var A_S : Array Integer [STREAM_SIZE_0 : Stream, VEC_SIZE_0 : Vector] Mem;
	Var B_S : Array Integer [STREAM_SIZE_0 : Stream, VEC_SIZE_0 : Vector] Mem;
	Var C_S : Array Integer [STREAM_SIZE_0 : Stream, VEC_SIZE_0 : Vector] Mem;
	Var D_S : Array Integer [STREAM_SIZE_0 : Stream, VEC_SIZE_0 : Vector] Mem;
	Var CycleVar_0 : Number ;
	Var CycleVar_1 : Number ;

Модификация циклов

Исходный текст программы	Параллельно-конвейерная форма

For i := 0 to N-1 do
Begin
com1 := a_s[i] * b_s[i];
c_s[i] := com1] / d_s[i];
End;
	For I:= 0 To STREAM_SIZE_0 - 1 Step 1 While 1 Do
	BEGIN
	For CycleVar_0:= 0 To VEC_SIZE_0 - 1 Step 1 While 1 Do
	BEGIN
	COM1[CycleVar_0] := A_S[I , CycleVar_0] * B_S[I , CycleVar_0];
	C_S[CycleVar_0] := COM1[CycleVar_0]/ D_S[I, CycleVar_0];
	END;
	END;

Виды редукции производительности

Редукция производительности программы – это пропорциональное сокращение производительности во всех без исключения фрагментах информационного графа задачи с **возможным** сокращением аппаратных затрат на реализацию вычислительной структуры.
Основное правило редукции производительности: при выполнении редукции производительности производительность сокращается **всегда и, в ряде определенных случаев**, сокращаются аппаратные затраты.

- редукция производительности по подграфам;

- редукция производительности по разрядности;

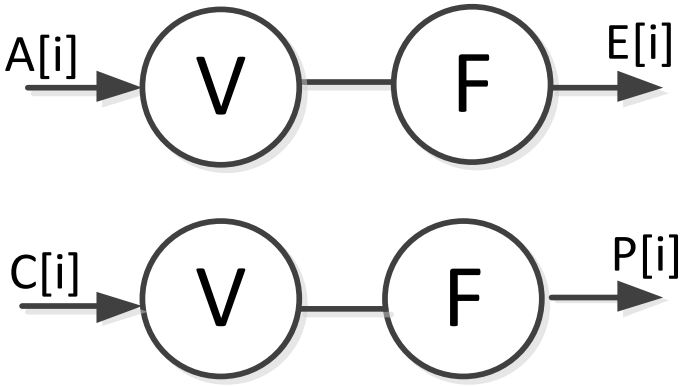
- редукция производительности по функциональным устройствам;

- редукция производительности по каналам данных;

- редукция производительности по частоте/скважности.

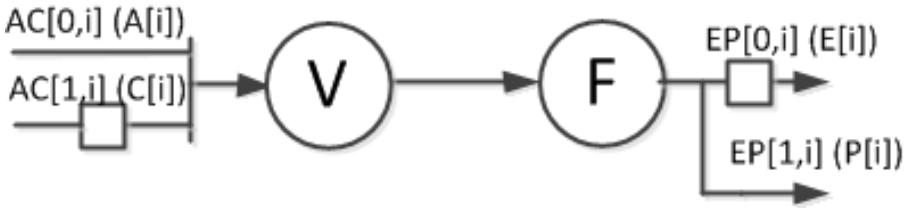
Редукция производительности по подграфам

```
Const X = 1;
Var a, e, c, p : Array Integer [N : Stream] Mem;
Var I : Number;
Cadr ExpReduction;
    For I := 0 to N-1 do
      begin
          #Reduction of Graph X
                E[i] := F(V(A[i]));
                P[i] := F(V(C[i]));
          #EndReduction;
      end;
EndCadr;
```



X = 1

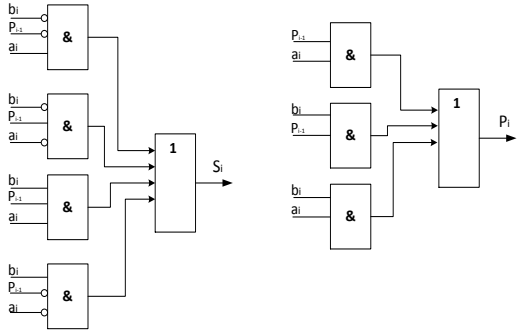
```
Const X = 2;
Var AC, EP : Array Integer [2: Vector, N : Stream]
Mem;
Var Com1 : Array Integer [N : Stream] Com;
Var I, J : Number;
Cadr ExpReduction;
  For J :=0 to 1 do
    For I := 0 to N-1 do
      begin
        EP[j,i] := F(V(Com1[i]));
         If j=0 then
           Com1[i] := AC[0,i]
          Else
           Com1[i] := AC[1,i]
      end;
EndCadr;
```



X = 2

Редукция производительности по разрядности

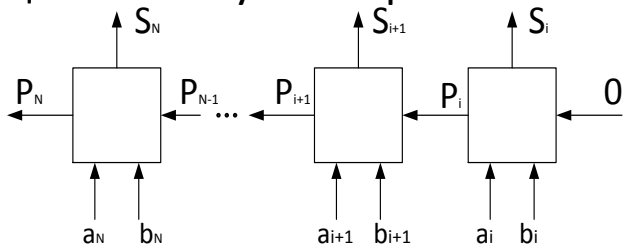
Логическая схема одноразрядного сумматора с переносом




```
SubCadr OneSectionAdd (in : a,b,Pin; out : S, Pout);
var a,b,Pout,S : [1:bit vector] Com;
var Pin,b1,b2,b3,b4,b5,b6,b7 : Boolean Com;

b1:=(not(b)) and (not(Pin)) and (a);
b2:=(not(b)) and (Pin) and (not(a));
b3:=(b) and (Pin) and (a);
b4:=(b) and (not(Pin)) and (not(a));

  S:=b1 or b2 or b3 or b4;

b5:=(Pin) and (a);
b6:=(b) and (Pin);
b7:=(b) and (a);

  Pout:=b5 or b6 or b7;
EndSubCadr;
```

Фрагмент программы описывающей одноразрядный сумматор с переносом

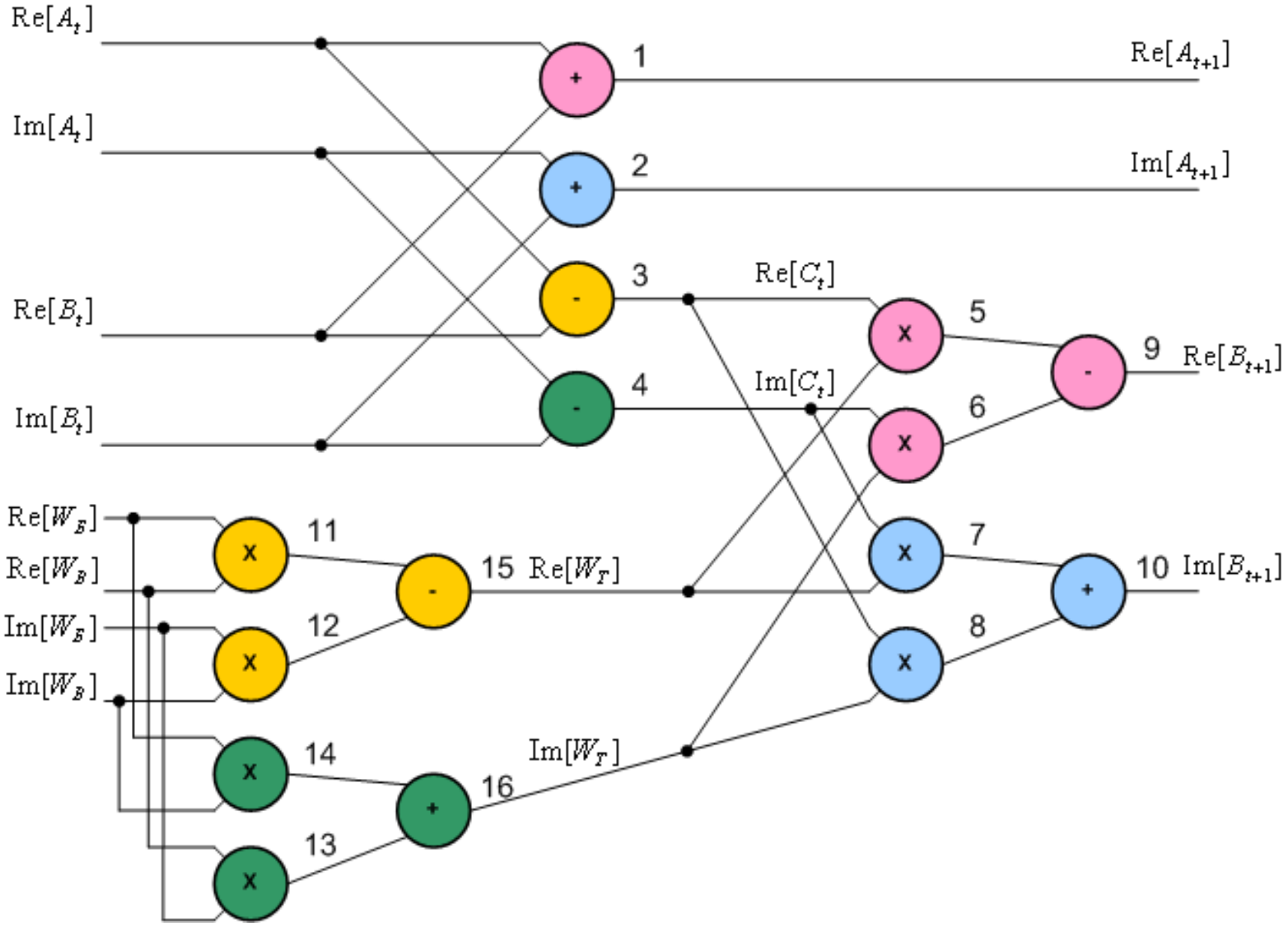
Логическая схема N-разрядного секционного сумматора




```
SubCadr NSectionAdd (in :a,b,N; out : Sout, P);
var N : Integer Com;
var a,b,Pout,S: [N:bit vector] Com;
var Sout : [N:bit vector] Com;
var P,b1,b2,b3,b4,b5,b6,b7 : Boolean Com;
var i,j : Number;

  OneSectionAdd (a[0],b[0],0,S[0],Pout[0]);
  For i:= 1 to N-1 do
    OneSectionAdd (a[i],b[i],Pout[i-1],S[i],Pout[i]);

  For j:= 0 to N-1 do
    Sout[j]:=S[j];
  Sout[N]:=Pout[N-1];
  P:=Pout[N-1];
EndSubCadr;
```

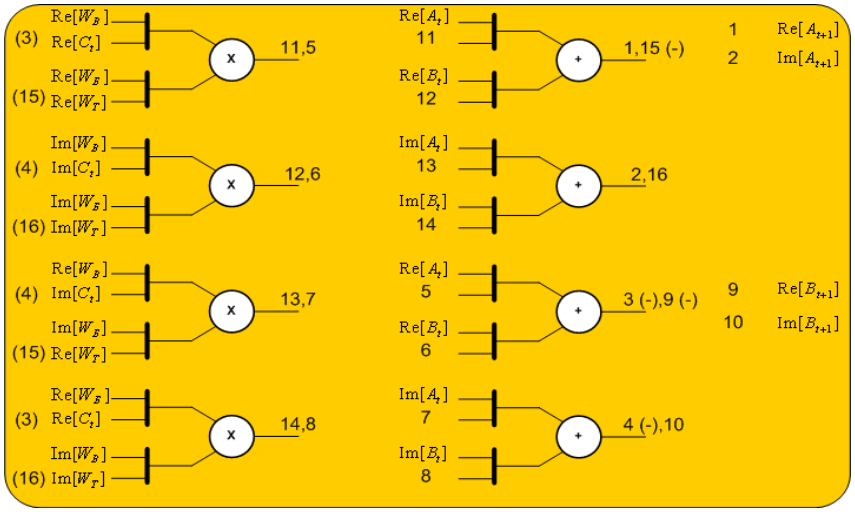
Фрагмент программы описывающей N-разрядный секционный сумматор

ИНФОРМАЦИОННЫЙ ГРАФ БАЗОВОЙ ОПЕРАЦИИ БПФ

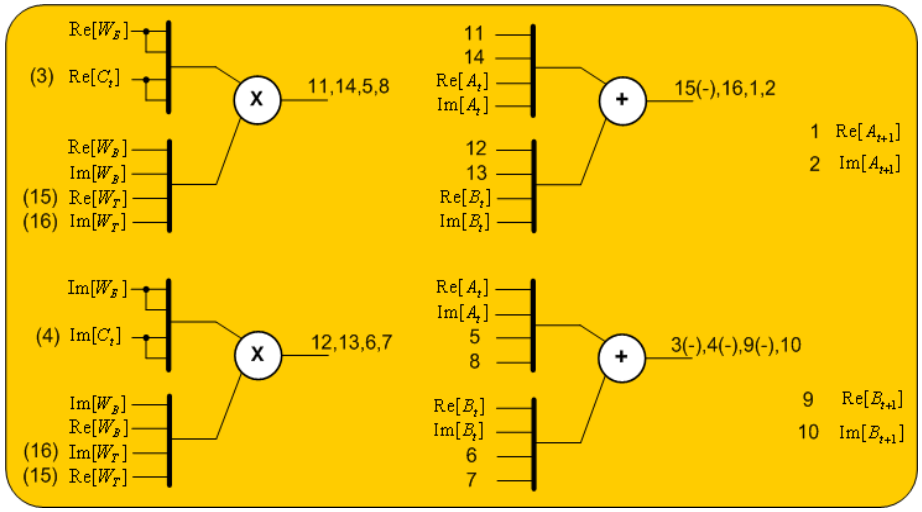


РЕДУКЦИЯ БАЗОВОЙ ОПЕРАЦИИ БПФ

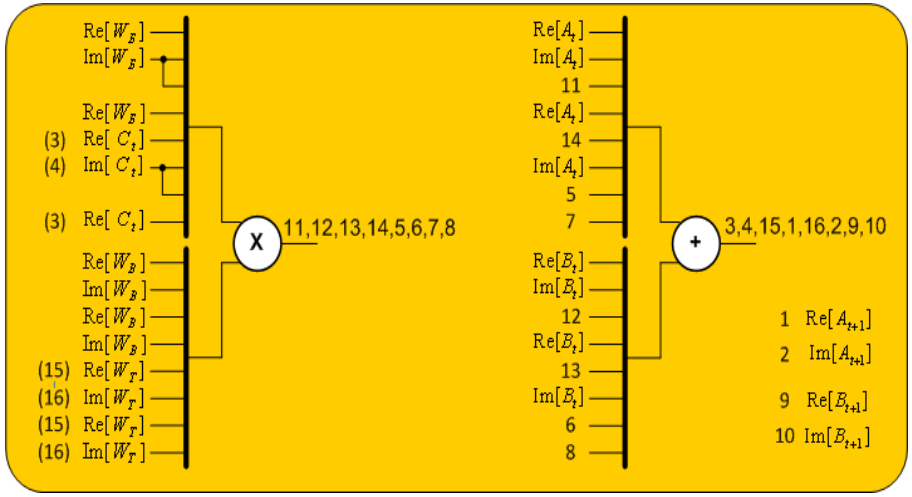
С КОЭФФИЦИЕНТОМ 2



С КОЭФФИЦИЕНТОМ 4



С КОЭФФИЦИЕНТОМ 8



Редукция производительности по устройствам

```
  subcadr BaseNode(in: In_A, In_B, In_C, In_D, In_F, In_G, In_E, In_M, In_N, In_K; out:
Out_X, Out_Y);
    var In_A, In_B, In_C, In_D, In_F, In_G, In_E, In_M, In_N, In_K, Out_X, Out_Y: Integer
Com;
    var com1, com2, com3, com4, com5, com6, com7, com8, com9: Integer Com;
    #Reduction of device 2;
      com1 := In_A * In_B;      com2 := In_C * In_D;
      com5 := com1 + com2;   com3 := In_F * In_G;
      com4 := In_E * In_M;      com7 := com3 - com4;
      com6 := In_N + com5;    com8 := com6 * com7;
      com9 := In_K + com7;
      Out_X := com8;
      Out_Y := com9;
    #EndReduction;
  EndSubCadr;

  Cadr Cadr1;
    for i := 0 to N1-1 do
    BaseNode(a[i], b[i], c[i], d[i], f[i], f[i], f[i], a[i], b[i], c[i], d[i], g[i]);
  EndCadr;
End_Program.
```

Редукция производительности по устройствам

Текст программы языке Colamo после применения редукции

```
SubCadr BASENODE( In: IN_A, IN_B, IN_C, IN_D, IN_F, IN_G, IN_E,
IN_M, IN_N, IN_K; Out: OUT_X, OUT_Y ) ;
   Var IN_A, IN_B, IN_C, IN_D, IN_F, IN_G, IN_E, IN_M, IN_N, IN_K,
OUT_X, OUT_Y, COM1, COM2, COM3, COM4, COM5, COM6, COM7,
COM8, COM9 : Integer Com;
   BaseCom ( IN_A , IN_B , IN_C , IN_D , IN_F , IN_G , IN_E , IN_M ,
IN_N , IN_K , OUT_X , OUT_Y  ) ;
EndSubCadr;

Cadr CADR1;
   For I:= 0 To STREAM_CONST_0 - 1 Step 1 While 1  Do
   BEGIN
   For CyrcleVar_0:= 0 To NEW_SIZE_0 - 1 Step 1 While 1  Do
   BEGIN
   BASENODE ( A[I , CycleVar_0 ] , B[I , CycleVar_0 ] , C[I , CycleVar_0 ]
, C[I , CycleVar_0 ] , F[I , CycleVar_0 ] , F[I , CycleVar_0 ] , F[I ,
CycleVar_0 ] , A[I , CycleVar_0 ] , B[I , CycleVar_0 ] , C[I , CycleVar_0 ] ,
D[I , CycleVar_0 ] , G[I , CycleVar_0 ]  ) ;
   END;
   END;
EndCadr;
End_Program.
```

До применения метода редукции в задаче было задействовано 9 функциональных устройств. После применения метода редукции с коэффициентом 2, задействовано осталось 5 устройств (3 умножителя, 2 блока АЛУ).

Достоинства разрабатываемой технологии ресурсонезависимого прикладного программного обеспечения ВСГТ

1. **Единое языковое пространство** для программирования всех узлов вычислительной системы гибридного типа.

2. **Автоматическая синхронизация** информационных потоков вычислительной структуры.

3. **Ресурсонезависимое программирование** – при изменении конфигурации вычислительной системы гибридного типа прикладные программы только перетранслируются.

4. **Прикладные программы могут портироваться** на различные архитектуры и конфигурации вычислительной системы.

5. **Сокращение времени** программирования и отладки до 1-3 месяцев.

Недостаток: ухудшение читабельности программы

БЛАГОДАРЮ ЗА ВНИМАНИЕ !