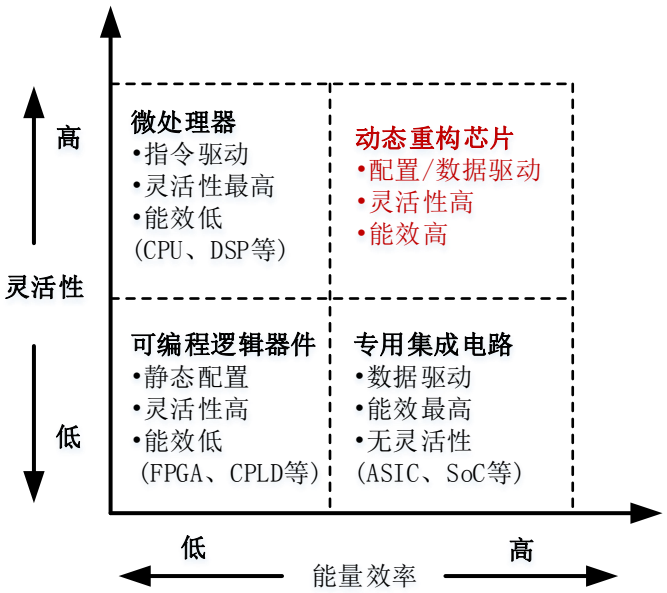
动态可重构密码芯片 SPEC

版本号	人员	时间	备注
V0.1	吴有余	2015/12/08	

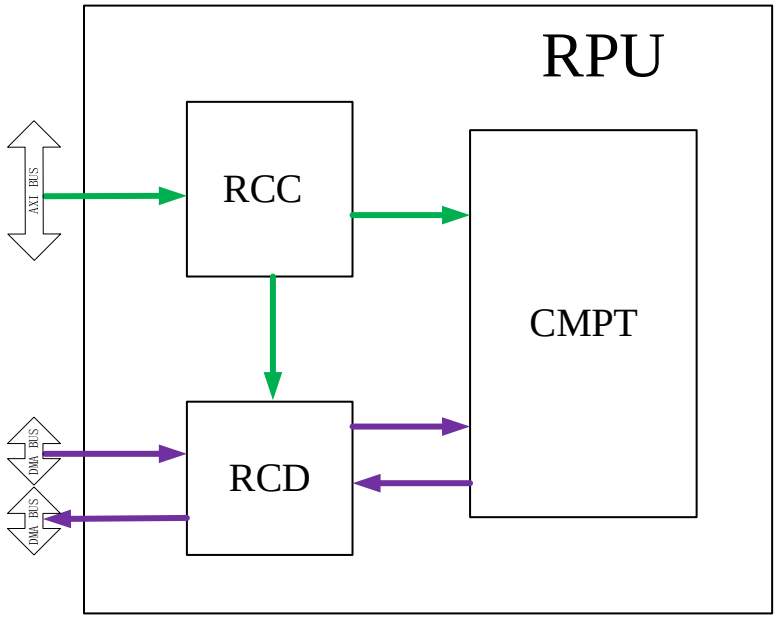
1. 项目背景

近年来，信息系统的安全形势变得越来越严峻，对密码处理器芯片的灵活性、速度、功耗和安全性等提出了更为苛刻的要求。当前的密码处理器芯片从体系结构和设计方法上来说主要可分为两大类。第一大类是专用集成电路（ASIC）的实现方式，第二类是指令集结构微处理器（ISAP）的实现方式。ASIC 实现方式往往会针对算法做出优化，故而运算速度远超 ISAP 实现方式，但其设计完成后，硬件结构不能改变，无法满足现有密码应用对灵活性的要求，若被破解只能废弃。ISAP 实现方式的密码处理器最大的优势在功能灵活性上，灵活性的取得往往是以牺牲能量效率为代价的，即此类处理器很难克服能量效率不高的重要缺陷。可重构密码处理器可以在以上两种处理器之间取得平衡，从而实现面向应用的最优折衷方案。



2. RPU 系统设计

结构框图：



功能说明：

RCC(Reconfigurable Controller)，可重构配置控制器模块，接收端口送入的配置参数信息，经解析后，将计算需要的配置参数送入 CMPT 模块，数据控制模块需要的送入 RCD 模块。

RCD(Reconfigurable Datapath)，可重构数据控制模块，接收外部接口送入的待处理数据，根据 RCC 送入参数配置信息，将数据送入 CMPT 对应接口进行处理，同时将 CMPT 处理后的结果输出至外部接口。

CMPT(Conputing Controller)，可重构计算模块，接收 RCC 送入的配置信息，经过初始化完成后，接收 RCD 送入的待处理数据，计算完成后，再将

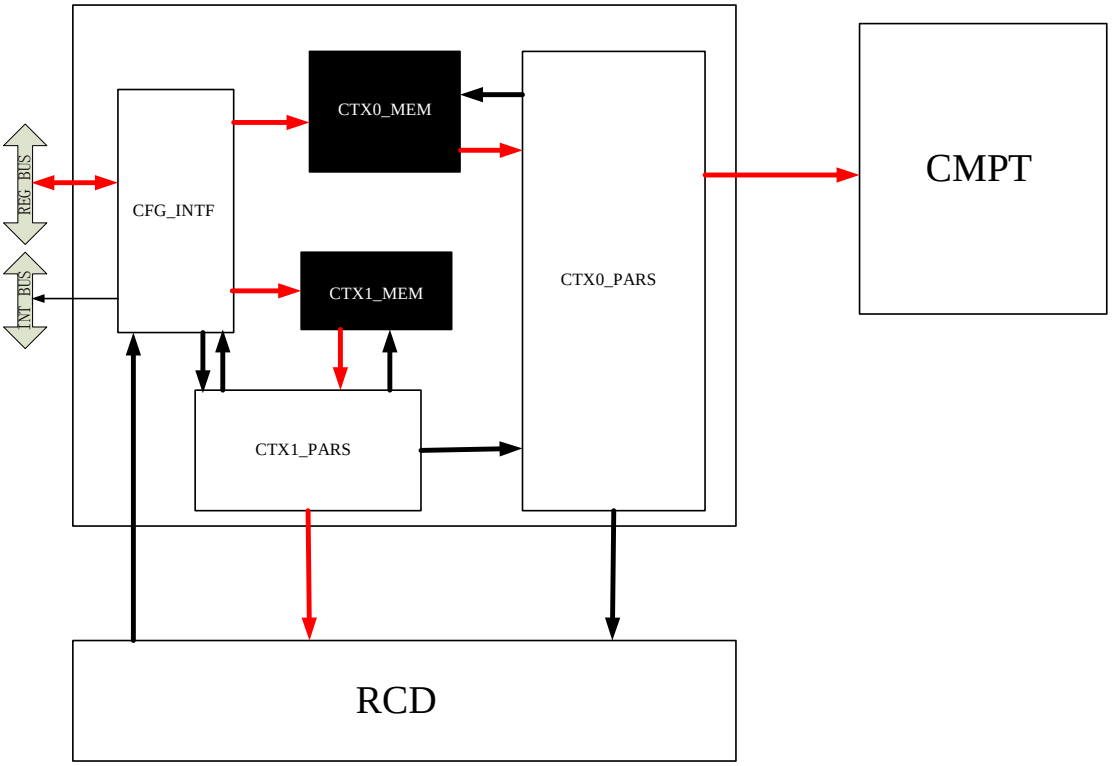
2.1 AXI Interface

结构框图

功能说明

2.2 Reconfigurable Controller

2.2.1 结构框图



该模块主要接收外部送入的配置参数信息，并存储到内部的存储器(CTX0_MEM 和 CTX1_MEM)中。外部配置寄存器启动算法后，再从存储器中将算法信息读出，并进行解析，解析后将算法配置参数送入对应模块(CMPT 和 RCD)。

CFG_INTF 模块用于接口处理，完成参数的存储，主要功能有：a，从外部寄存器接口总线上接收算法参数，并根据地址配置，将算法参数写入对应的存储器；b，接收 RCD 送入的反馈数据，并根据反馈配置，将反馈数据写入对应的参数存储器；c，接收送入的状态信息，如有中断发生，将中断送出外部的中断总线。

CTX0_MEM 模块主要用来存储底层运算和运算数据通路的配置参数，主要功能有：a，接收 CFG_INTF 模块的参数写控制信号，并将各类参数存储到对应

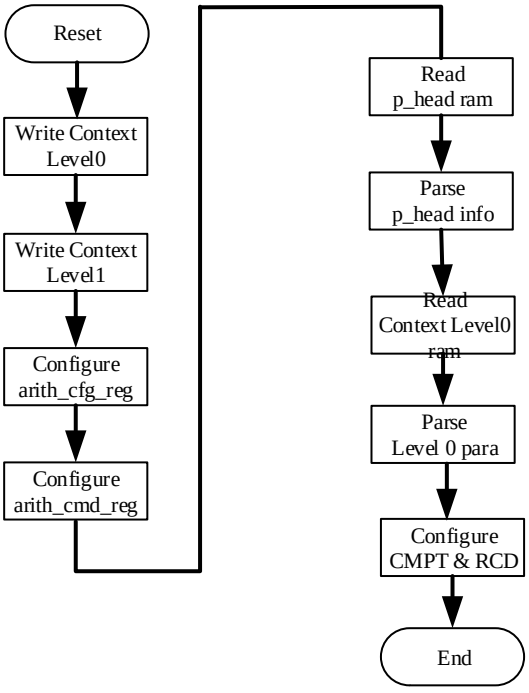
存储器；b，接收 CTX0_PARS 模块的读控制信号，并将读取到的参数信息送入 CTX0_PARS 模块。

CTX1_MEM 模块主要用来存储算法的上层配置信息，主要功能有：a，接收 CFG_INTF 模块的参数写控制信号，并将参数存储到存储器；b，接收 CTX1_PARS 模块的读控制信号，并将读取到的参数信息送入 CTX1_PARS 模块。

CTX0_PARS 模块主要产生 CMPT 所需要的各类参数信息，主要功能有：a，接收 CTX1_PARS 的控制信息，读取对应的 CTX0_MEM 中的参数；b，根据 CTX1_PARS 的控制信息，将读取的 CTX0_MEM 中的参数送入 CMPT 模块。

CTX1_PARS 模块主要是解析 CTX1_MEM 中读取到的参数信息，主要功能有：a，接收 CFG_INTF 模块的算法配置寄存器的控制，然后发出读取 CTX1_MEM 的控制；b，将从 CTX1_MEM 中读取对应的算法配置信息，进行解析，将解析后的算法参数控制信息送入 CTX0_PARS 模块，结果输出的控制信号至 RCD 模块，以及将反馈配置送入 CFG_INTF 模块；

算法参数加载过程如下所示：



复位后，第 1 步，外部通过 CFG_INTF 接口完成 CTX0_MEM 中的参数注入；第 2 步，完成 CTX1_MEM 中的参数注入；第 3 步，配置 CFG_INTF 中的算法配置寄存器；第 4 步，向 CFG_INTF 中的算法命令寄存器发出算法启动命令；第 5 步，CTX1_PARS 模块从 CTX1_MEM 中读取算法配置信息；第 6 步，将算法配

置信息进行解析，将解析后的算法控制信号送入 CTX0_PARS，以及完成 RCD 模块的配置；第 7 步，CTX0_PARS 模块接收到算法参数控制信号，发出读取 CTX0_MEM 操作；第 8 步，CTX0_PARS 模块接收到算法参数，送入 CMPT 模块，完成了所有的参数加载。

2.2.2 接口信号

Name	Bits	From/To	Description
CLK_I	1	全局	时钟信号
RST_N_I	1	全局	复位信号，低有效
REG_WEN_I	1	接口	寄存器接口，写使能
REG_WADDR_I	16	接口	寄存器接口，写地址
REG_WDAT_I	32	接口	寄存器接口，写数据
REG_REN_I	1	接口	寄存器接口，读使能
REG_RADDR_I	16	接口	寄存器接口，读地址
REG_RDAT_O	32	接口	寄存器接口，读返回数据
IRQ_N_O	1	接口	中断输出，低有效
RCD_PARA_RDY_O	1	RCD	参数配置完成指示信号
RCD_CBC_EN_O	1	RCD	CBC 模式使能信号
RCD_CHAN_EN_O	4	RCD	RCD 通道使能信号，每 1bit 对应 1 个通道
RCD_CHAN_MODE_O	8	RCD	数据模式，每 2bit 对应 1 个通道 2’b00，128bit 有效, 2’b01，64bit 有效； 2’b10，32bit 有效 其余，暂时保留
RCD_OFIFO_MAP_O	8	RCD	输出 FIFO 选择，每 2bit 对应 1 个通道； 2’b00，选择 ofifo 0 输出数据； 2’b01，选择 ofifo 1 输出数据； 2’b10，选择 ofifo2 输出数据; 2’b11，选择 ofifo3 输出数据.
RCD_FBK_EN_O	4	RCD	反馈通道使能信号，每 1 个通道对应 1bit 配置
RCD_FBK_FADDR_O	32	RCD	反馈通道读起始地址配置，每 1 个通道对应 8bit
RCD_FBK_AOFFSET_O	16	RCD	反馈通道读偏移地址配置，每 1 个通道对应 4bit
RCD_FBK_DSIZE_O	16	RCD	反馈通道读数据次数配置，每 1 个通道对应 4bit
RCD_FBK_WEN0_I	1	RCD	至 RCC 的反馈数据写使能 0

RCD_FBK_WEN1_I	1	RCD	至 RCC 的反馈数据写使能 1
RCD_FBK_WEN2_I	1	RCD	至 RCC 的反馈数据写使能 2
RCD_FBK_WEN3_I	1	RCD	至 RCC 的反馈数据写使能 3
RCD_FBK_WDAT0_I	32	RCD	至 RCC 的反馈数据 0
RCD_FBK_WDAT1_I	32	RCD	至 RCC 的反馈数据 1
RCD_FBK_WDAT2_I	32	RCD	至 RCC 的反馈数据 2
RCD_FBK_WDAT3_I	32	RCD	至 RCC 的反馈数据 3
RCD_IMD_EN_O	4	RCD	IMD 写使能
RCD_IMD_DAT0_O	32	RCD	IMD 数据线 0
RCD_IMD_DAT1_O	32	RCD	IMD 数据线 1
RCD_IMD_DAT2_O	32	RCD	IMD 数据线 2
RCD_IMD_DAT3_O	32	RCD	IMD 数据线 3
RCD_IMD_RSP_I	4	RCD	IMD 接收响应信号
CMPT_R0_RC_CFG_O	128	CMPT	CMPT 中阵列第 0 行的 RC 参数配置
CMPT_R1_RC_CFG_O	128	CMPT	CMPT 中阵列第 1 行的 RC 参数配置
.....			
CMPT_R27_RC_CFG_O	128	CMPT	CMPT 中阵列第 27 行的 RC 参数配置
CMPT_R0_CR_CFG_O	168	CMPT	CMPT 中阵列第 0 行的 CR 参数配置
CMPT_R1_CR_CFG_O	168	CMPT	CMPT 中阵列第 1 行的 CR 参数配置
.....			
CMPT_R27_CR_CFG_O	168	CMPT	CMPT 中阵列第 27 行的 CR 参数配置
CMPT_SB_CEN_N_O	1	CMPT	CMPT 中阵列的 SBOX 片选，低有效
CMPT_SB_WEN_N_O	1	CMPT	CMPT 中阵列的 SBOX 写使能，低有效
CMPT_SB_WADDR_O	5	CMPT	CMPT 中阵列的 SBOX 写地址
CMPT_SB_WDAT_O	1024	CMPT	CMPT 中阵列的 SBOX 写数据
CMPT_R1_PU_CFG_O	352	CMPT	CMPT 中阵列第 1 行 PU 配置参数
CMPT_R3_PU_CFG_O	352	CMPT	CMPT 中阵列第 3 行 PU 配置参数
.....			
CMPT_R27_PU_CFG_O	352	CMPT	CMPT 中阵列第 27 行 PU 配置参数
CMPT_R0_IMD1_O	128	CMPT	CMPT 中阵列第 0 行常量
CMPT_R1_IMD1_O	128	CMPT	CMPT 中阵列第 1 行常量
.....			
CMPT_R27_IMD1_O	128	CMPT	CMPT 中阵列第 27 行常量
CMPT_ADN_CFG_O	847	CMPT	CMPT 中 ADN 模块的配置参数
CMPT_GPRF_CFG_O	752	CMPT	CMPT 中 GPRF 控制器的配置参数
CMPT_RARA_SSWIT_O	4	CMPT	CMPT 中参数自流水更换使能
CMPT_SSWT_CFG_O	12	CMPT	CMPT 中参数自流水更换控制配置参数
CMPT_PSWIT_REQ_I	28	CMPT	CMPT 中阵列各行参数自流水切换请求，每 1bit 对应 1 行

2.2.3 模块设计

2.2.3.1. CFG_INTF

存储器空间分配

Name	Offset	W/R	SIZE	Description
reg	0x0000~0x00ff	----	32*64	寄存器空间
p_rc_ram	0x0100~0x04ff	WR	128*64	RC 配置参数存储器
p_cr_ram	0x0500~0x0aff	WR	168*64	CR 配置参数存储器
p_pu_ram	0x0c00~0x117f	WR	352*32	PU BENES 配置参数存储器
p_sb_ram	0x1180~0x217f	WR	256*32*4	SBOX 配置参数存储器 4 个 S 盒，每个大小为 256*32
p_imd0_ram	0x2180~0x297f	WR	128*128	IMD0 配置参数存储器
p_imd1_ram	0x2980~0x317f	WR	128*128	IMD1 配置参数存储器
p_adn_ram	0x3180~0x335f	WR	960*4	ADN 配置参数存储器
p_gdp_ram	0x3380~034ff	WR	748*4	GPRF 控制器配置参数存储器
p_pkt_ram	0x3500~0x38ff	WR	32*256	算法参数包存储器

寄存器设置

Name	Offset	W/R	Reset Value	Description
ARITH_CFG_REG	0x00	WR	0x00000000	算法配置寄存器
ARITH_CMD_REG	0x04	WO	0x00000000	算法命令寄存器
ARITH_STAT_REG	0x08	RO	0x00000000	工作状态寄存器

算法配置寄存器的设置如下

ARITH_CFG_REG: 0x00

Name	Bits	W/R	Reset Value	Description
Reserved	[31:11]	—	0x0	Reserved
ARITH_ID_CFG	[10:8]	WR	0x0	算法 ID，用于和算法储存器中 ID

				比对，检测是否一致
ARITH_PKT_ADDR	[7:0]	WR	0x00	算法参数存储首地址

ARITH_CMD_REG

Name	Bits	W/R	Reset Value	Description
Reserved	[31:8]	—	0x0	Reserved
ARITH_CMD	[7:0]	WO	0x00	算法命令 0x10，算法启动配置命令； 0x20，RPU 软复位命令； 0x30，无数据输入启动阵列工作

ARITH_STAT_REG

Name	Bits	W/R	Reset Value	Description
LVL1_FSM_STAT	[3:0]	R	0x0	Level 1 参数解析状态机状态值
LVL0_ROW_FSM_STAT	[5:4]	R	0x0	Level 0 行参数加载状态机状态值
LVL0_SB_FSM_STAT	[7:6]	R	0x0	Level 0SBOX 加载状态机状态值
LVL0_IMD0_FSM_STAT	[9:8]	R	0x0	Level 0 IMD0 加载状态机状态值
LVL0_IMD1_FSM_STAT	[11:10]	R	0x0	Level 0 IMD1 加载状态机状态值
RCD_FSM_STAT	[14:12]	R	0x0	RCD 中状态机状态值
ARITH_CODE_ERR	[15]	R	0x0	算法编号配置与 P_Head 中值不一致
ARITH_CFG_RDY	[16]	R	0x0	算法参数配置完成指示。

接口信号

Name	Bits	From/To	Description
CLK_I	1	全局	时钟信号
RST_N_I	1	全局	复位信号，低有效
REG_WEN_I	1	接口	寄存器接口，写使能
REG_WADDR_I	16	接口	寄存器接口，写地址
REG_WDAT_I	32	接口	寄存器接口，写数据
REG_REN_I	1	接口	寄存器接口，读使能
REG_RADDR_I	16	接口	寄存器接口，读地址
REG_RDAT_O	32	接口	寄存器接口，读返回数据

IRQ_N_O	1	接口	中断输出，低有效
RC_RAM_WEN_N_O	1	CTX0_MEM	RC 参数存储器写使能
RC_RAM_WADDR_O	6	CTX0_MEM	RC 参数存储器写地址
RC_RAM_WDAT_O	128	CTX0_MEM	RC 参数存储器写数据
CR_RAM_WEN_N_O	1	CTX0_MEM	CR 参数存储器写使能
CR_RAM_WADDR_O	6	CTX0_MEM	CR 参数存储器写地址
CR_RAM_WDAT_O	168	CTX0_MEM	CR 参数存储器写数据
PU_RAM_WEN_N_O	1	CTX0_MEM	PU 参数存储器写使能
PU_RAM_WADDR_O	5	CTX0_MEM	PU 参数存储器写地址
PU_RAM_WDAT_O	352	CTX0_MEM	PU 参数存储器写数据
SB_RAM_WEN_N_O	1	CTX0_MEM	SBOX 参数存储器写使能
SB_RAM_WADDR_O	7	CTX0_MEM	SBOX 参数存储器写地址
SB_RAM_WDAT_O	256	CTX0_MEM	SBOX 参数存储器写数据
IMD0_RAM_WEN_N_O	1	CTX0_MEM	IMD0 参数存储器写使能
IMD0_RAM_WADDR_O	7	CTX0_MEM	IMD0 参数存储器写地址
IMD0_RAM_WDAT_O	128	CTX0_MEM	IMD0 参数存储器写数据
IMD1_RAM_WEN_N_O	1	CTX0_MEM	IMD1 参数存储器写使能
IMD1_RAM_WADDR_O	7	CTX0_MEM	IMD1 参数存储器写地址
IMD1_RAM_WDAT_O	128	CTX0_MEM	IMD1 参数存储器写数据
ADN_RAM_WEN_N_O	1	CTX0_MEM	ADN 参数存储器写使能
ADN_RAM_WADDR_O	2	CTX0_MEM	ADN 参数存储器写地址
ADN_RAM_WDAT_O	847	CTX0_MEM	ADN 参数存储器写数据
GDP_RAM_WEN_N_O	1	CTX0_MEM	GPRF 控制参数存储器写使能
GDP_RAM_WADDR_O	2	CTX0_MEM	GPRF 控制参数存储器写地址
GDP_RAM_WDAT_O	752	CTX0_MEM	GPRF 控制参数存储器写数据
PKT_RAM_WEN_N_O	1	CTX1_MEM	算法包参数存储器写使能
PKT_RAM_WADDR_O	8	CTX1_MEM	算法包参数存储器写地址
PKT_RAM_WDAT_O	32	CTX1_MEM	算法包参数存储器写数据
ARITH_CFG_REG_O	32	CTX1_PARS	算法参数配置寄存器
ARITH_CMD_EN_O	1	CTX1_PARS	算法命令寄存器使能
ARITH_CMD_REG_O	32	CTX1_PARS	算法命令寄存器
FBK_MODE_I	2	CTX1_PARS	2bit，反馈模式： 2‘b00,反馈至 IMD0； 2‘b01,反馈至 IMD1； 其他，暂时保留
FBK_MEM_FADDR_I	6	CTX1_PARS	反馈存储器首地址
ARITH_ID_ERR_I	1	CTX1_PARS	算法 ID 不一致错误
FBK_WEN_I	4	RCD	反馈数据写使能
FBK_WDAT_I	128	RCD	反馈写数据

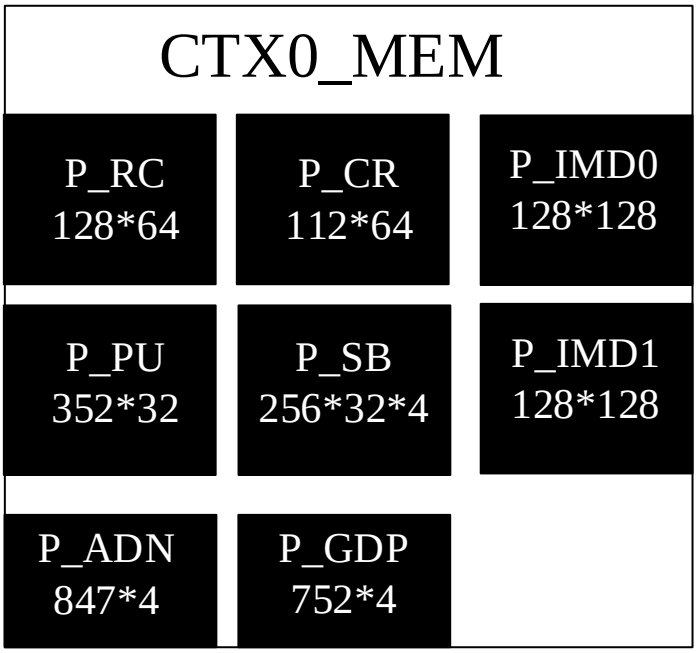
功能实现

CFG_INTF 主要利用寄存器接口，将接收到的参数根据地址的不同，写入不同的存储器，由于寄存器接口是统一的 32bit，而存储器接口位宽各不一样，因

此 CFG_INTF 中电路实现最主要需要利用计数器，来实现数据的拼接，将拼接后的数据写入参数存储器。

2.2.3.2. CTX0_MEM

结构框图



功能说明：

该模块主要存储底层单元配置参数。

RC，运算单元的配置参数；

CR，

接口信号

Name	Bits	From/To	Description
CLK_I	1	全局	时钟信号
RST_N_I	1	全局	复位信号，低有效
RC_RAM_WEN_N_I	1	CFG_INTF	RC 参数存储器写使能
RC_RAM_WADDR_I	6	CFG_INTF	RC 参数存储器写地址
RC_RAM_WDAT_I	128	CFG_INTF	RC 参数存储器写数据
CR_RAM_WEN_N_I	1	CFG_INTF	CR 参数存储器写使能
CR_RAM_WADDR_I	6	CFG_INTF	CR 参数存储器写地址
CR_RAM_WDAT_I	168	CFG_INTF	CR 参数存储器写数据

PU_RAM_WEN_N_I	1	CFG_INTF	PU 参数存储器写使能
PU_RAM_WADDR_I	5	CFG_INTF	PU 参数存储器写地址
PU_RAM_WDAT_I	352	CFG_INTF	PU 参数存储器写数据
SB_RAM_WEN_N_I	1	CFG_INTF	SBOX 参数存储器写使能
SB_RAM_WADDR_I	7	CFG_INTF	SBOX 参数存储器写地址
SB_RAM_WDAT_I	256	CFG_INTF	SBOX 参数存储器写数据
IMD0_RAM_WEN_N_I	1	CFG_INTF	IMD0 参数存储器写使能
IMD0_RAM_WADDR_I	7	CFG_INTF	IMD0 参数存储器写地址
IMD0_RAM_WDAT_I	128	CFG_INTF	IMD0 参数存储器写数据
IMD1_RAM_WEN_N_I	1	CFG_INTF	IMD1 参数存储器写使能
IMD1_RAM_WADDR_I	7	CFG_INTF	IMD1 参数存储器写地址
IMD1_RAM_WDAT_I	128	CFG_INTF	IMD1 参数存储器写数据
ADN_RAM_WEN_N_I	1	CFG_INTF	ADN 参数存储器写使能
ADN_RAM_WADDR_I	2	CFG_INTF	ADN 参数存储器写地址
ADN_RAM_WDAT_I	847	CFG_INTF	ADN 参数存储器写数据
GDP_RAM_WEN_N_I	1	CFG_INTF	GPRF 控制参数存储器写使能
GDP_RAM_WADDR_I	2	CFG_INTF	GPRF 控制参数存储器写地址
GDP_RAM_WDAT_O	752	CFG_INTF	GPRF 控制参数存储器写数据
RC_RAM_REN_N_I	1	CTX0_PARS	RC 参数存储器读使能
RC_RAM_RADDR_I	6	CTX0_PARS	RC 参数存储器读地址
RC_RAM_RDAT_O	128	CTX0_PARS	RC 参数存储器读返回数据
CR_RAM_REN_N_I	1	CTX0_PARS	CR 参数存储器读使能
CR_RAM_RADDR_I	6	CTX0_PARS	CR 参数存储器读地址
CR_RAM_RDAT_I	168	CTX0_PARS	CR 参数存储器读返回数据
PU_RAM_REN_N_I	1	CTX0_PARS	PU 参数存储器读使能
PU_RAM_RADDR_I	5	CTX0_PARS	PU 参数存储器读地址
PU_RAM_RDAT_O	352	CTX0_PARS	PU 参数存储器读返回数据
SB_RAM_REN_N_I	1	CTX0_PARS	SBOX 参数存储器读使能
SB_RAM_RADDR_I	7	CTX0_PARS	SBOX 参数存储器读地址
SB_RAM_RDAT_O	256	CTX0_PARS	SBOX 参数存储器读返回数据
IMD0_RAM_REN_N_I	1	CTX0_PARS	IMD0 参数存储器读使能
IMD0_RAM_RADDR_I	7	CTX0_PARS	IMD0 参数存储器读地址
IMD0_RAM_RDAT_O	128	CTX0_PARS	IMD0 参数存储器读返回数据
IMD1_RAM_REN_N_I	1	CTX0_PARS	IMD1 参数存储器读使能
IMD1_RAM_RADDR_I	7	CTX0_PARS	IMD1 参数存储器读地址
IMD1_RAM_RDAT_O	128	CTX0_PARS	IMD1 参数存储器读返回数据
ADN_RAM_REN_N_I	1	CTX0_PARS	ADN 参数存储器读使能
ADN_RAM_RADDR_I	2	CTX0_PARS	ADN 参数存储器读地址
ADN_RAM_RDAT_O	847	CTX0_PARS	ADN 参数存储器读返回数据
GDP_RAM_REN_N_I	1	CTX0_PARS	GPRF 控制参数存储器读使能
GDP_RAM_RADDR_I	2	CTX0_PARS	GPRF 控制参数存储器读地址
GDP_RAM_RDAT_O	752	CTX0_PARS	GPRF 控制参数存储器读返回数据

2.2.3.3. CTX1_MEM

结构框图

接口信号

Name	Bits	From/To	Description
CLK_I	1	全局	时钟信号
RST_N_I	1	全局	复位信号，低有效
PKT_RAM_WEN_N_I	1	CFG_INTF	算法包参数存储器写使能
PKT_RAM_WADDR_I	8	CFG_INTF	算法包参数存储器写地址
PKT_RAM_WDAT_I	32	CFG_INTF	算法包参数存储器写数据
PKT_RAM_REN_N_I	1	CTX1_PARS	算法包参数存储器读使能
PKT_RAM_RADDR_I	8	CTX1_PARS	算法包参数存储器读地址
PKT_RAM_RDAT_O	32	CTX1_PARS	算法包参数存储器读数据

功能实现

存储器报文结构如下表所示：

配置位置	含义
Word0[31:29]	3bit，算法编号
Word0[28:27]	2bit，暂时保留
Word0[26]	1bit，算法是否更换行参数
Word0[25:21]	5bit，Word0[26]如果为高，算法包需要调用的行参数类型数目
Word0[20:17]	4bit，列有效，每一 bit 对应一列

Word0[16]	1bit，算法是否更换 SBOX
Word0[15]	1bit，Word0[18]，单行 RC 是否使用同一种 SBOX
Word0[14:13]	2bit，Word0[17]如果为高，则使用同一种 SBOX，SBOX 的编号
Word0[14:7]	8bit，Word0[17]如果为低，表示同一行使用了不同的 SBOX [16:15]，RC 列地址 0 使用的 SBOX 编号； [14:13]，RC 列地址 1 使用的 SBOX 编号； [12:11]，RC 列地址 2 使用的 SBOX 编号； [10:9]，RC 列地址 3 使用的 SBOX 编号；
Word0[6:5]	2bit，算法使用 ADN 参数编号
Word0[4:3]	2bit，算法使用 GPRF Datapath 参数编号
Word0[2]	1bit，算法是否需要更换参数
Word0[1]	算法参数更换类型 1’b0：自流水更换参数 1’b1：全局统一更换参数
Word0[0]	1bit，算法是否有反馈
Word1[31:25]	7bit，暂时保留
Word1[24]	1bit，算法是否更换 IMD1
Word1[23]	IMD1 使用方式： 1’b0，送至 CMPT ROW 端口； 1’b1，送至 GPRF 端口
Word1[22:17]	6bit，IMD1 MEM 读起始地址
Word1[16:12]	5bit，IMD1 MEM 读次数
Word1[11:10]	2bit，如果 Word1[23]为 1’b0，则表示 IMD1 在 RCA 中使用的方式； 2‘b00, 1 行使用 128bit IMD1 数据； 2’b01, 1 行使用 64bit IMD1 数据； 2’b10, 1 行使用 32bit IMD1 数据； 2’b11, 暂时保留；

	注，如果没有使用全部 128bit，例如 64bit，则 1 行 IMD1 有 RCA 中 2 行使用。
Word1[9:5]	5bit，IMD1 在 RCA 阵列中使用的第一个行编号
Word1[4:0]	5bit，IMD1 在 RCA 阵列中使用间隔。例如如果值为 5'b0，则表示每一行都需要使用 IMD1；如果为 5'b1，则表示 1 行间隔 1 行使用 IMD1；以此类推。
Word2[31:9]	23bit，暂时保留
Word2[8]	1bit，算法是否更换 IMD0
Word2[7:2]	6bit，IMD0 起始地址
Word2[1:0]	2bit，IMD0 MEM 读次数
Word3[31]	1bit，暂时保留
Word3[30:26]	5bit，该类行参数配置在 RCA 阵列中映射起始行编号；
Word3[25:22]	4bit，该类行参数配置在 RCA 阵列中映射的次数；
Word3[21:18]	4bit，该类行参数配置在 RCA 阵列中间隔；例如如果值为 4'b0，则表示连续的行使用此行参数配置；如果为 5'b1，则表示 1 行间隔 1 行使用此行参数配置；以此类推。
Word3[17:12]	6bit，CR 地址
Word3[11]	1bit，是否为 PU 配置
Word3[10:6]	5bit，如果 Word2 [11]有效，则表示 PU 地址,否则保留
Word3[5:0]	6bit，表示 RC 地址
.....	.....
Word(m-3) [31:30]	2bit，反馈模式： 2'b00,反馈至 IMD0； 2'b01,反馈至 IMD1； 其他，暂时保留
Word(m-3) [29:24]	6bit，反馈在参数 MEM 中的起始地址
Word(m-3)[23:20]	4bit，4 个反馈通道使能
Word(m-3)[19:12]	8bit，暂时保留
Word(m-3)[11:0]	12bit，参数自流水更换配置。

~~Word(m-3)[19:0]~~	~~20bit，4 个反馈通道在 RCA 中的反馈行地址~~
Word(m-2)[31:24]	8bit，反馈通道 0 的数据在 GPRF 中起始地址
Word(m-2) [23:16]	8bit，反馈通道 1 的数据在 GPRF 中起始地址
Word(m-2)[15:8]	8bit，反馈通道 2 的数据在 GPRF 中起始地址
Word(m-2) [7:0]	8bit，反馈通道 3 的数据在 GPRF 中起始地址
Word(m-1)[31:28]	4bit，反馈通道 0 在 GPRF 中访问地址累加值，如果为固定地址反馈方式，该值为 0
Word(m-1)[27:24]	4bit，反馈通道 0 在 GPRF 中一次访问数据次数
Word(m-1) [23:20]	4bit，反馈通道 1 在 GPRF 中访问地址累加值，如果为固定地址反馈方式，该值为 0
Word(m-1) [19:16]	4bit，反馈通道 1 在 GPRF 中一次访问数据次数
Word(m-1)[15:12]	4bit，反馈通道 2 在 GPRF 中访问地址累加值，如果为固定地址反馈方式，该值为 0
Word(m-1)[11:8]	4bit，反馈通道 2 在 GPRF 中一次访问数据次数
Word(m-1) [7:4]	4bit，反馈通道 3 在 GPRF 中访问地址累加值，如果为固定地址反馈方式，该值为 0
Word(m-1) [3:0]	4bit，反馈通道 3 在 GPRF 中一次访问数据次数
Word[31:24]	8bit，暂时保留
Word[23:0]	24bit，RCA 结果输出 OFIFO 控制参数
Word(m) [31:20]	12bit，暂时保留
Word(m) [19:16]	4bit，RCD 的 4 个通道使能
Word(m) [15:8]	8bit，RCD 的 4 个通道数据模式 [15:14]，2’b00，128bit 有效；2’b01，64bit 有效； 2’b10，32bit 有效；暂时保留 其余类似。
Word(m) [7:0]	8bit，RCD 的 4 个输入通道对应输出通道配置 [7:6]，输入通道 0 对应输出通道位置配置 [5:4]，输入通道 1 对应输出通道位置配置 [3:2]，输入通道 2 对应输出通道位置配置

	[1:0]，输入通道 3 对应输出通道位置配置

算法中 m 值为，需要配置的行类数目+ 1(Head) + 1(IMD1)+1(IMD0) + 3(FBK) + 1(RCD)，即如果需要有 n 行的参数配置，则 m 值为 n+7。

2.2.3.4. CTX0_PARS

接口信号

Name	Bits	From/To	Description
CLK_I	1	全局	时钟信号
RST_N_I	1	全局	复位信号，低有效
LVL0_RC_STAT_O	2	CFG_INTF	行参数加载状态机值输出
LVL0_SB_FSM_STAT_O	2	CFG_INTF	SBOX 参数加载状态机值输出
LVL0_IMD0_FSM_STAT_O	2	CFG_INTF	IMD0 参数加载状态机值输出
LVL0_IMD1_FSM_STAT_O	2	CFG_INTF	IMD1 参数加载状态机值输出
CTX1_CFG_END_I	1	CTX1_PARS	CTX1 参数配置解析完成，电平信号
PARA_GLB_SWIT_REQ_I	1	CTX1_PARS	算法参数全局切换请求，电平信号
PARA_SELF_SWIT_REQ_I	1	CTX1_PARS	算法参数自流水切换
ROW_PARA_SWIT_EN_I	1	CTX1_PARS	行参数切换使能
ROW_PARA_RC_EN_I	4	CTX1_PARS	行 RC 使能，4bit 对应 1 行中 4 个 RC
ROW_PARA_CFG_EN_I	1	CTX1_PARS	行参数存储器读使能
ROW_RC_RADDR_I	6	CTX1_PARS	P_RC_RAM 读地址
ROW_CR_RAADR_I	6	CTX1_PARS	P_CR_RAM 读地址
RCA_ROW_FADDR_I	5	CTX1_PARS	行参数在 RCA 阵列中映射的首个行地址
RCA_ROW_MAP_NUM_I	4	CTX1_PARS	行参数在 RCA 阵列中行映射的次数
RCA_ROW_MAP_OFFSET_I	4	CTX1_PARS	行参数在 RCA 阵列中行映射的偏移地址
PU_PARA_EN_I	1	CTX1_PARS	P_PU_RAM 读使能
PU_PARA_RADDR_I	5	CTX1_PARS	P_PU_RAM 读地址
SBOX_SWIT_EN_I	1	CTX1_PARS	SBOX 参数切换使能
SBOX_SWIT_TYPE_I	1	CTX1_PARS	SBOX 切换类型
SBOX_SWIT_CFG_I	8	CTX1_PARS	SBOX 读操作配置
ADN_PARA_EN_I	1	CTX1_PARS	P_ADN_RAM 读使能

ADN_PARA_RADDR_I	2	CTX1_PARS	P_ADN_RAM 读地址
GPRF_PARA_EN_I	1	CTX1_PARS	P_GDP_RAM 读使能
GPRF_PARA_RADDR_I	5	CTX1_PARS	P_GDP_RAM 读地址
IMD1_SWIT_EN_I	4	CTX1_PARS	P_IMD1_RAM 读使能
IMD1_PARA_RD_NUM_I	5	CTX1_PARS	P_IMD1_RAM 读次数
IMD1_PARA_FADDR_I	6	CTX1_PARS	P_IMD1_RAM 读地址
IMD1_RCA_MODE_I	2	CTX1_PARS	P_IMD1_RAM 读数据使用方式 2'b00, 1 行使用 128bit IMD1 数据； 2'b01, 1 行使用 64bit IMD1 数据； 2'b10, 1 行使用 32bit IMD1 数据； 2'b11, 暂时保留； 注，如果没有使用全部 128bit，例如 64bit，则 1 行 IMD1 有 RCA 中 2 行使用
IMD1_ROW_FADDR_I	5	CTX1_PARS	IMD1 在 RCA 阵列中使用的第一个行编号
IMD1_ROW_OFFSET_I	5	CTX1_PARS	IMD1 在 RCA 阵列中使用间隔。例如如果值为 5'b0，则表示每一行都需要使用 IMD1；如果为 5'b1，则表示 1 行间隔 1 行使用 IMD1；以此类推。
IMD0_SWIT_EN_I	1	CTX1_PARS	算法是否更换 IMD0
IMD0_PARA_RD_NUM_I	2	CTX1_PARS	P_IMD0_RAM 读次数
IMD0_PARA_FADDR_I	6	CTX1_PARS	P_IMD0_RAM 起始地址
PARA_GLB_SWIT_EN_I	1	CTX0_PARS	行参数统一切换使能
PARA_SELF_SWIT_CFG_I	12	CTX1_PARS	参数自流水切换配置
RCA_OUTPUT_CFG_I	24	CTX1_PARS	RCA 阵列结果输出配置
RC_RAM_REN_N_O	1	CTX0_MEM	RC 参数存储器写使能
RC_RAM_RADDR_O	6	CTX0_MEM	RC 参数存储器写地址
RC_RAM_RDAT_I	128	CTX0_MEM	RC 参数存储器写数据
CR_RAM_REN_N_O	1	CTX0_MEM	CR 参数存储器写使能
CR_RAM_RADDR_O	6	CTX0_MEM	CR 参数存储器写地址
CR_RAM_RDAT_I	168	CTX0_MEM	CR 参数存储器写数据
PU_RAM_REN_N_O	1	CTX0_MEM	PU 参数存储器写使能
PU_RAM_RADDR_O	5	CTX0_MEM	PU 参数存储器写地址
PU_RAM_RDAT_I	352	CTX0_MEM	PU 参数存储器写数据
SB_RAM_REN_N_O	1	CTX0_MEM	SBOX 参数存储器写使能
SB_RAM_RADDR_O	7	CTX0_MEM	SBOX 参数存储器写地址
SB_RAM_RDAT_I	256	CTX0_MEM	SBOX 参数存储器写数据
IMD0_RAM_REN_N_O	1	CTX0_MEM	IMD0 参数存储器写使能
IMD0_RAM_RADDR_O	7	CTX0_MEM	IMD0 参数存储器写地址
IMD0_RAM_RDAT_I	128	CTX0_MEM	IMD0 参数存储器写数据
IMD1_RAM_REN_N_O	1	CTX0_MEM	IMD1 参数存储器写使能

IMD1_RAM_RADDR_O	7	CTX0_MEM	IMD1 参数存储器写地址
IMD1_RAM_RDAT_I	128	CTX0_MEM	IMD1 参数存储器写数据
ADN_RAM_REN_N_O	1	CTX0_MEM	ADN 参数存储器写使能
ADN_RAM_RADDR_O	2	CTX0_MEM	ADN 参数存储器写地址
ADN_RAM_RDAT_I	847	CTX0_MEM	ADN 参数存储器写数据
GDP_RAM_REN_N_O	1	CTX0_MEM	GPRF 控制参数存储器写使能
GDP_RAM_RADDR_I	2	CTX0_MEM	GPRF 控制参数存储器写地址
GDP_RAM_RDAT_O	752	CTX0_MEM	GPRF 控制参数存储器写数据
RCA_ROW_SWIT_REQ_I	28	CMPT	RCA 行参数自切换请求
PARA_SELF_SWIT_CFG_O	12	CMPT	参数自流水切换配置
RCA_OUTPUT_CFG_O	24	CMPT	RCA 阵列结果输出配置
CMPT_R0_RC_CFG_O	128	CMPT	CMPT 中阵列第 0 行的 RC 参数配置
CMPT_R1_RC_CFG_O	128	CMPT	CMPT 中阵列第 1 行的 RC 参数配置
.....			
CMPT_R27_RC_CFG_O	128	CMPT	CMPT 中阵列第 27 行的 RC 参数配置
CMPT_R0_CR_CFG_O	168	CMPT	CMPT 中阵列第 0 行的 CR 参数配置
CMPT_R1_CR_CFG_O	168	CMPT	CMPT 中阵列第 1 行的 CR 参数配置
.....			
CMPT_R27_CR_CFG_O	168	CMPT	CMPT 中阵列第 27 行的 CR 参数配置
CMPT_SB_CEN_N_O	1	CMPT	CMPT 中阵列的 SBOX 片选，低有效
CMPT_SB_WEN_N_O	1	CMPT	CMPT 中阵列的 SBOX 写使能，低有效
CMPT_SB_WADDR_O	5	CMPT	CMPT 中阵列的 SBOX 写地址
CMPT_SB_WDAT_O	1024	CMPT	CMPT 中阵列的 SBOX 写数据
CMPT_R1_PU_CFG_O	352	CMPT	CMPT 中阵列第 1 行 PU 配置参数
CMPT_R3_PU_CFG_O	352	CMPT	CMPT 中阵列第 3 行 PU 配置参数
.....			
CMPT_R27_PU_CFG_O	352	CMPT	CMPT 中阵列第 27 行 PU 配置参数
CMPT_R0_IMD1_O	128	CMPT	CMPT 中阵列第 0 行常量
CMPT_R1_IMD1_O	128	CMPT	CMPT 中阵列第 1 行常量
.....			
CMPT_R27_IMD1_O	128	CMPT	CMPT 中阵列第 27 行常量
CMPT_ADN_CFG_O	847	CMPT	CMPT 中 ADN 模块的配置参数

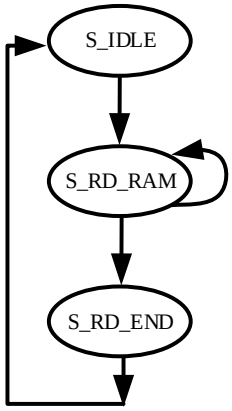
CMPT_GPRF_CFG_O	752	CMPT	CMPT 中 GPRF 控制器的配置参数
CMPT_RARA_SSWIT_O	4	CMPT	CMPT 中参数自流水更换使能
CMPT_SSWT_CFG_O	12	CMPT	CMPT 中参数自流水更换控制配置参数
CMPT_PSWIT_REQ_I	28	CMPT	CMPT 中阵列各行参数自流水切换请求，每 1bit 对应 1 行
GPRF_IMD1_WEN_N_O	1	CMPT	CMPT 中 GPRF 写使能，低有效
GPRF_IMD1_WADDR_O	6	CMPT	CMPT 中 GPRF 写地址
GPRF_IMD1_WDAT_O	128	CMPT	CMPT 中 GPRF 写数据
RCD_IMD_EN_O	4	RCD	IMD 写使能
RCD_IMD_DAT0_O	128	RCD	IMD 数据线 0
RCD_IMD_DAT1_O	128	RCD	IMD 数据线 1
RCD_IMD_DAT2_O	128	RCD	IMD 数据线 2
RCD_IMD_DAT3_O	128	RCD	IMD 数据线 3
RCD_IMD_RSP_I	4	RCD	IMD 接收响应信号

功能实现

在 CTX0_PARS 模块中有 2 套 CMPT 参数配置寄存器(CMPT_*_CURR_REG 和 CMPT_*_NEXT_REG，*号为不同的参数名)，当前正在使用的参数为 CMPT_*_CURR_REG，待切换使用的参数为 CMPT_*_NEXT_REG，切换时将 CMPT_*_NEXT_REG 中的值送入 CMPT_*_CURR_REG。

CTX0_PARS 模块功能 a，接收 CTX1_PARS 的控制信息，读取对应的 CTX0_MEM 中的参数的功能实现，是由行 RC、行 CR、PU、SBOX、IMD0、IMD1 的 5 个状态机控制来实现。

状态机的设计如下图所示(在设计代码中需要为每一个状态机单独取名)：



S_LV1_IDLE:空闲状态，在该状态中，状态机处于空闲状态，如果存储器读使能，则状态机进入 S_RD_RAM 状态；

S_RD_RAM: 读参数存储器状态，在该状态中发出参数存储器的读使能和读地址，同时对读次数进行计数，当读使能信号无效后，进入 S_RD_END 状态，否则一致保持在该状态。

S_RD_END: 参数存储器读操作完成状态，下一周期直接进入 S_IDLE 状态。

下面分别对这 5 个状态机的条件信号进行说明：

行 RC 读状态机和行 CR 读状态机跳转条件和行 RC 一样：当接收到 CTX1_PARS 模块送入的 ROW_PARA_SWIT_EN_I 行参数切换使能后，进入到行 RC 参数存储器读操作状态，当 ROW_PARA_SWIT_EN_I 无效后，进入 S_RD_END 状态。

PU 参数读状态机：当接收到 CTX1_PARS 模块送入的 PU_PARA_EN_I 行参数切换使能后，进入到行参数存储器读操作状态，当 PU_PARA_EN_I 无效后，进入 S_RD_END 状态。

SBOX 读状态机：当接收到 CTX1_PARS 模块送入的 SBOX 参数切换使能 SBOX_SWIT_EN_I，并根据读操作配置，进行 SBOX 读，当完成所有的 S 盒数据（32 次）读操作后，进入到 S_RD_END 状态。

IMD0 读状态机：当接收到 CTX1_PARS 模块送入的 IMD0_SWIT_EN_I 使能信号后，进入到 IMD0 读状态，读地址由 IMD0_PARA_FADDR_I 配置而得，超过 1 次后，在此地址上累加 1，当完成 IMD0_PARA_RD_NUM_I 配置的读操作次数后，进入到 S_RD_END 状态，IMD0 的数据是送入 RCD 模块。

IMD1 读状态机：当接收到 CTX1_PARS 模块送入的 IMD1_SWIT_EN_I 使

能信号后，进入到 IMD1 读状态，读地址由 IMD1_PARA_FADDR_I 配置而得，超过 1 次后，在此地址上累加 1，当完成 IMD1_PARA_RD_NUM_I 配置的读操作次数后，进入到 S_RD_END 状态，IMD1 的数据是送入 CMPT 模块。

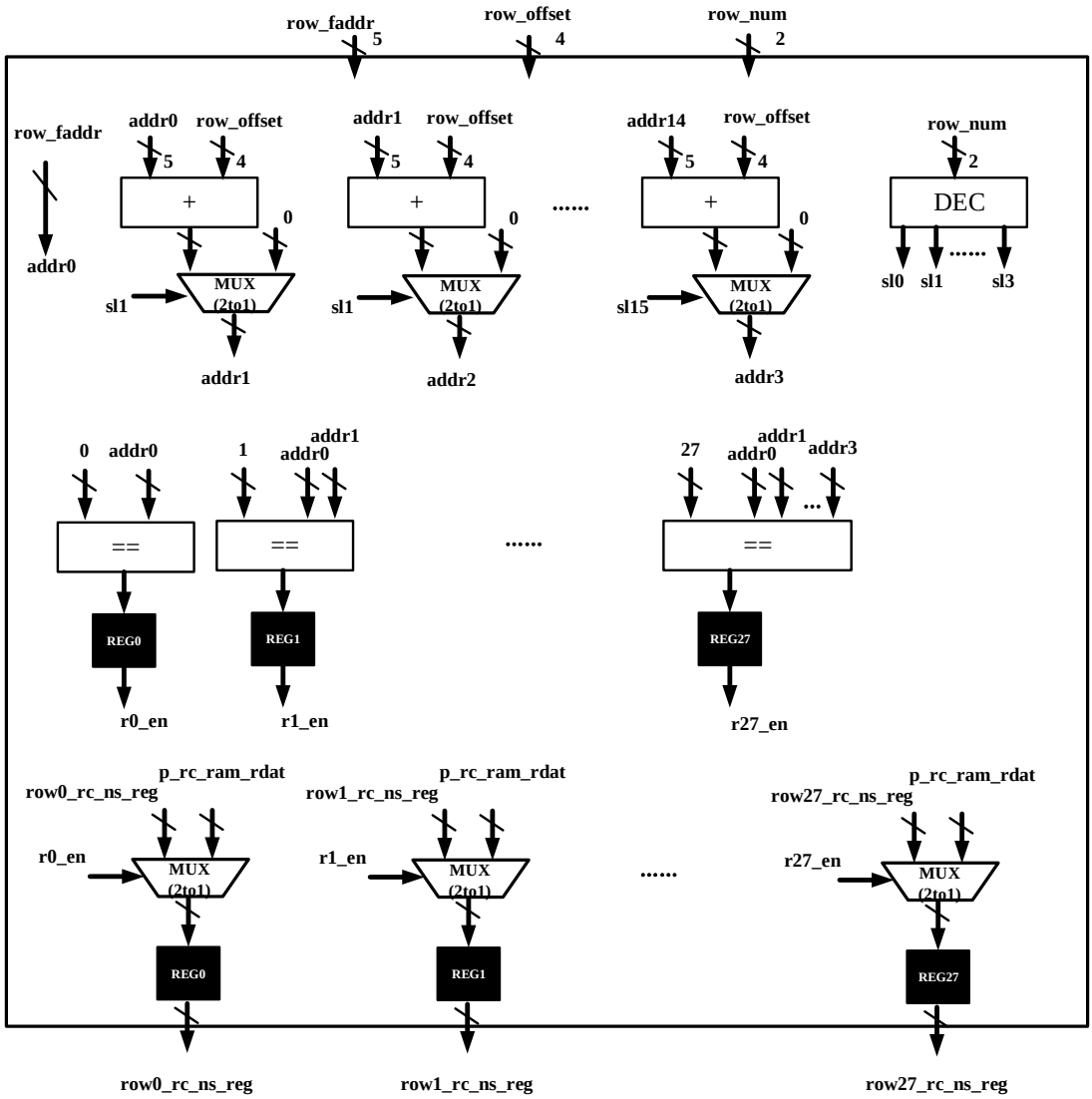
CTX0_PARS 模块功能 b，根据 CTX1_PARS 的控制信息，将读取的 CTX0_MEM 中的参数送入 CMPT 模块。针对不同的参数类型，在 CMPT 中映射的控制方式是不同的。

RC 参数、CR 参数、IMD0 采用的是映射次数+映射首地址+映射行地址偏移来实现的；SBOX、PU 参数是对所有的单元进行相同的配置。

在 CTX0_PARS 中设有 2 套参数寄存器(当前参数寄存器和待切换参数寄存器)，当发生参数更换请求时，将带切换参数寄存器的值赋值给当前参数寄存器。

以 RC 参数为例来说明，CTX0_PARS 根据 CTX1_PARS 送入的 ROW_RC_RADDR_I 和 ROW_PARA_CFG_EN_I 信号，发出读 CTX0_MEM 中 P_RC_RAM 的读控制；下一周期得到 P_RC_RAM 返回的值，将返回值与 ROW_PARA_RC_EN_I 操作后(4bit 对应 128bit，每一 bit 对应 32bit，如果 4bit 中某 1bit 为 0，则该 bit 对应的 32bit 输出为 0)，送给 RCA_ROW_FADDR_I 所对应的行(例如 RCA_ROW_FADDR_I 值为 0，则送给 R0_RC_CFG_NXT_REG，如果 RCA_ROW_FADDR_I 为 3，则送给 R3_RC_CFG_NXT_REG)。如果 RCA_ROW_MAP_NUM_I 的值不是 1，则共有 RCA_ROW_MAP_NUM_I 行的待切换参数寄存器被赋值，分别为 RCA_ROW_FADDR_I，(RCA_ROW_FADDR_I+RCA_ROW_MAP_OFFSET_I)，(RCA_ROW_FADDR_I+RCA_ROW_MAP_OFFSET_I*2)，。。。

当 CTX1_PARS 送入的 PARA_GLB_SWIT_EN_I 为 0 (即采用自流水参数切换方式)，则 CMPT 送入的 RCA_ROW_SWIT_REQ_I[27:0]中对应的行请求 bit 位有效后，待切换参数寄存器中的值赋值给当前使用参数寄存器；如果 PARA_GLB_SWIT_EN_I 为 1(即采用行参数统一切换方式)，则在所有的参数读取完成后，将待切换参数寄存器中的值赋值给当前使用参数寄存器。



2.2.3.5. CTX1_PARS

接口信号

Name	Bits	From/To	Description
CLK_I	1	全局	时钟信号
RST_N_I	1	全局	复位信号，低有效
PKT_RAM_REN_N_O	1	CTX1_MEM	算法包参数存储器读使能
PKT_RAM_RADDR_O	6	CTX1_MEM	算法包参数存储器读地址
PKT_RAM_RDAT_I	32	CTX1_MEM	算法包参数存储器读数据
ARITH_CFG_REG_I	32	CFG_INTF	算法参数配置寄存器
ARITH_CMD_EN_I	1	CFG_INTF	算法命令寄存器使能
ARITH_CMD_REG_I	32	CFG_INTF	算法命令寄存器
FBK_MODE_O	2	CFG_INTF	2bit，反馈模式： 2‘b00,反馈至 IMD0；

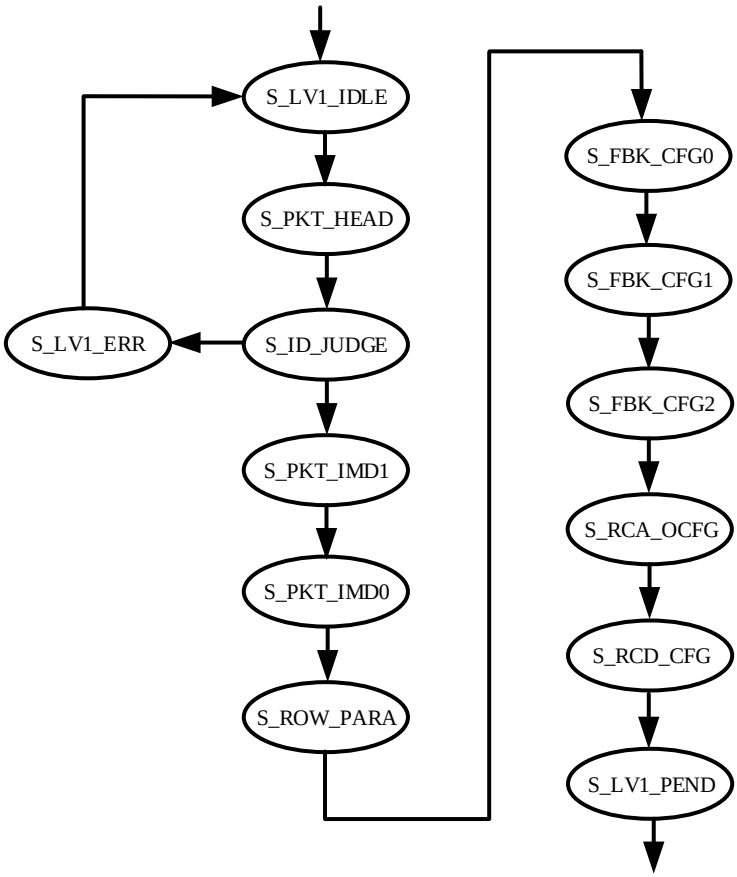
			2‘b01,反馈至 IMD1； 其他，暂时保留
FBK_MEM_FADDR_O	6	CFG_INTF	反馈存储器首地址
ARITH_ID_ERR_O	1	CFG_INTF	算法 ID 不一致错误，脉冲信号
LV1_FSM_STATE_O	4	CFG_INTF	状态机值输出
CTX1_CFG_END_O	1	CTX0_PARS	CTX1 参数配置解析完成，电平信号
PARA_GLB_SWIT_REQ_O	1	CTX0_PARS	算法参数全局切换请求，电平信号
PARA_SELF_SWIT_REQ_O	1	CTX0_PARS	算法参数自流水切换
ROW_PARA_SWIT_EN_O	1	CTX0_PARS	行参数切换使能
ROW_PARA_RC_EN_O	4	CTX0_PARS	行 RC 使能，4bit 对应 1 行中 4 个 RC
ROW_PARA_CFG_EN_O	1	CTX0_PARS	行参数存储器读使能
ROW_RC_RADDR_O	6	CTX0_PARS	P_RC_RAM 读地址
ROW_CR_RAADR_O	6	CTX0_PARS	P_CR_RAM 读地址
RCA_ROW_FADDR_O	5	CTX0_PARS	行参数在 RCA 阵列中映射的首个行地址
RCA_ROW_MAP_NUM_O	4	CTX0_PARS	行参数在 RCA 阵列中行映射的次数
RCA_ROW_MAP_OFFSET_O	4	CTX0_PARS	行参数在 RCA 阵列中行映射的偏移地址
PU_PARA_EN_O	1	CTX0_PARS	P_PU_RAM 读使能
PU_PARA_RADDR_O	5	CTX0_PARS	P_PU_RAM 读地址
SBOX_SWIT_EN_O	1	CTX0_PARS	SBOX 参数切换使能
SBOX_SWIT_TYPE_O	1	CTX0_PARS	SBOX 切换类型
SBOX_SWIT_CFG_O	8	CTX0_PARS	SBOX 读操作配置
ADN_PARA_EN_O	1	CTX0_PARS	P_ADN_RAM 读使能
ADN_PARA_RADDR_O	2	CTX0_PARS	P_ADN_RAM 读地址
GPRF_PARA_EN_O	1	CTX0_PARS	P_GDP_RAM 读使能
GPRF_PARA_RADDR_O	5	CTX0_PARS	P_GDP_RAM 读地址
IMD1_SWIT_EN_O	4	CTX0_PARS	P_IMD1_RAM 读使能
IMD1_PARA_RD_NUM_O	5	CTX0_PARS	P_IMD1_RAM 读次数
IMD1_PARA_FADDR_O	6	CTX0_PARS	P_IMD1_RAM 读地址
IMD1_RCA_MODE_O	2	CTX0_PARS	P_IMD1_RAM 读数据使用方式 2‘b00, 1 行使用 128bit IMD1 数据； 2’b01, 1 行使用 64bit IMD1 数据； 2’b10, 1 行使用 32bit IMD1 数据； 2’b11, 暂时保留； 注，如果没有使用全部 128bit，例如 64bit，则 1 行 IMD1 有 RCA 中 2 行使用
IMD1_ROW_FADDR_O	5	CTX0_PARS	IMD1 在 RCA 阵列中使用的第一个行编号
IMD1_ROW_OFFSET_O	5	CTX0_PARS	IMD1 在 RCA 阵列中使用间隔。例如如果值为 5’b0，则表示每一行都需要使用 IMD1；如果为 5’b1，则表示 1

			行间隔 1 行使用 IMD1；以此类推。
IMD0_SWIT_EN_O	1	CTX0_PARS	算法是否更换 IMD0
IMD0_PARA_RD_NUM_O	6	CTX0_PARS	P_IMD0_RAM 起始地址
IMD0_PARA_FADDR_O	2	CTX0_PARS	P_IMD0_RAM 读次数
PARA_GLB_SWIT_EN_O	1	CTX0_PARS	行参数统一切换使能
PARA_SELF_SWIT_CFG_O	12	CTX0_PARS	行参数自流水切换配置
RCA_OUTPUT_CFG_O	24	CTX0_PARS	RCA 阵列结果输出配置
RCD_CHAN_EN_O	4	RCD	RCD 通道使能信号
RCD_CHAN_MODE_O	8	RCD	数据模式，每 2bit 对应 1 个通道 2’b00，128bit 有效, 2’b01，64bit 有效； 2’b10，32bit 有效 其余，暂时保留
RCD_OFIFO_MAP_O	8	RCD	输出 FIFO 选择，每 2bit 对应 1 个通道； 2’b00，选择 ofifo 0 输出数据； 2’b01，选择 ofifo 1 输出数据； 2’b10，选择 ofifo2 输出数据; 2’b11，选择 ofifo3 输出数据.
RCD_FBK_EN_O	4	RCD	反馈通道使能信号，每 1 个通道对应 1bit 配置
RCD_FBK_FADDR_O	32	RCD	反馈通道读起始地址配置，每 1 个通道对应 8bit
RCD_FBK_AOFFSET_O	16	RCD	反馈通道读偏移地址配置，每 1 个通道对应 4bit
RCD_FBK_DSIZE_O	16	RCD	反馈通道读数据次数配置，每 1 个通道对应 4bit

功能实现

CTX1_PARS 功能的实现主要是根据状态机控制，来实现对 CTX1_MEM 中存储的算法配置信息的读取和解析，并将解析后的控制信息输出值 CTX0_PARS 和 RCD 模块。

状态机的设计如下图所示：



功能说明：

该模块主要根据配置，从 Level1 MEM 中取出算法包信息，并进行解析，将解析后的信息送入 Level0 解析模块、RCD 模块、CMPT 模块。

状态说明：

S_LV1_IDLE:空闲状态，在该状态中，状态机处于空闲状态，如果收到算法启动命令，则状态机进入 S_PKT_HEAD 状态；

S_PKT_HEAD: 算法信息头(CTX1_MEM 存储器中的 WORD0)读取状态，在该状态中，发出读取算法参数存储器，读地址由 CFG_INT 中 ARITH_CFG_REG 寄存器配置而来；

S_ID_JUDGE：算法参数 ID 判断状态，在该状态中得到从算法存储器返回的数据 WORD0，并解析，判断参数中算法 ID 是否和 ARITH_CFG_REG 中 ID 值是否一致，如果不一致，则输入参数错误，下一状态跳转到 S_LV1_ERR 状态，否则跳转到 S_PKT_IMD1 状态。同时将 WORD0 中其余参数控制信息输出；

S_PKT_IMD1：IMD1(CTX1_MEM 存储器中的 WORD1)配置控制参数读取状态，

在该状态中继续发出读取算法参数存储器读使能，读地址由上一次读操作加 1 得到；

S_PKT_IMD0：IMD0(CTX1_MEM 存储器中的 WORD2)配置控制参数读取状态，在该状态中继续发出读取算法参数存储器读使能，读地址由上一次读操作加 1 得到；

S_ROW_PARA:行配置控制参数读取状态在该状态中继续发出读取算法参数存储器读使能，读地址由上一次读操作加 1 得到，并对读行配置控制参数次数进行累加，当累加的次数与 WORD0 中配置的次数相同时，就跳转到 S_FBK_CFG0 状态，否则继续保持在该状态。

S_FBK_CFG0：如果有反馈，反馈配置字 0(CTX1_MEM 存储器中的 WORD (m-3))读取状态，下一周期直接跳转到 S_FBK_CFG1 状态；

S_FBK_CFG1：如果有反馈，反馈配置字 1(CTX1_MEM 存储器中的 WORD (m-2))读取状态，下一周期直接跳转到 S_FBK_CFG2 状态；

S_FBK_CFG2：如果有反馈，反馈配置字 2(CTX1_MEM 存储器中的 WORD (m-1))读取状态，下一周期直接跳转到 S_RCA_OCFG 状态；

S_RCA_OCFG:输出结果结果参数配置，下一周期直接跳转到 S_RCD_CFG。

S_RCD_CFG: RCD 通道配置字(CTX1_MEM 存储器中的 WORD m)读取状态，下一周期直接跳转到 S_LV1_PEND 状态；

S_LV1_PEND: CTX1_MEM 中信息读取完成，下一周期直接跳转到 S_RCD_CFG 状态。

2.3 Reconfigurable Datapath

结构框图

功能说明

2.3.1 DMA Input Interface

结构框图

功能说明

2.3.2 DMA Output Interface

结构框图

功能说明

2.3.3 Input FIFO

结构框图

功能说明

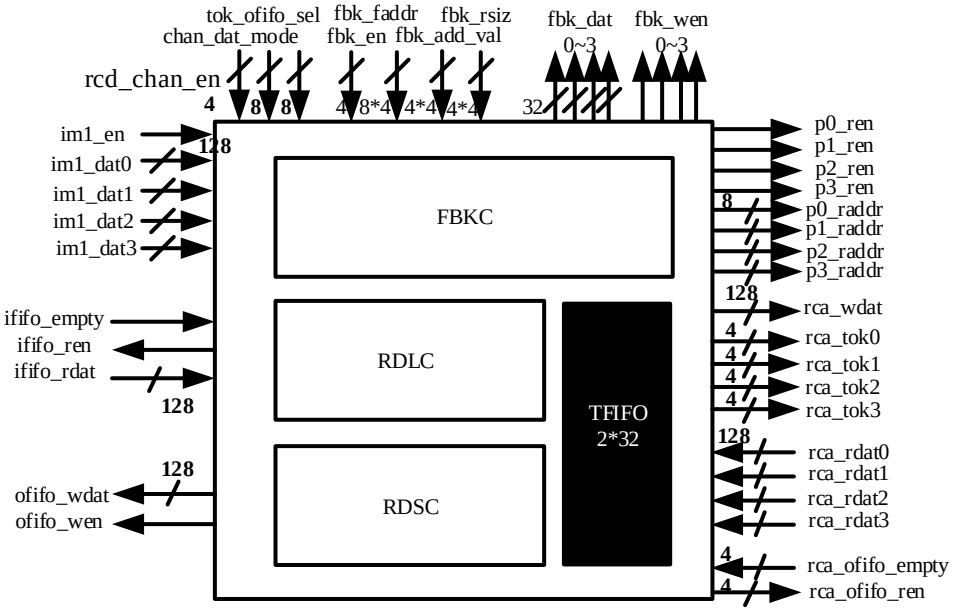
2.3.4 Output FIFO

结构框图

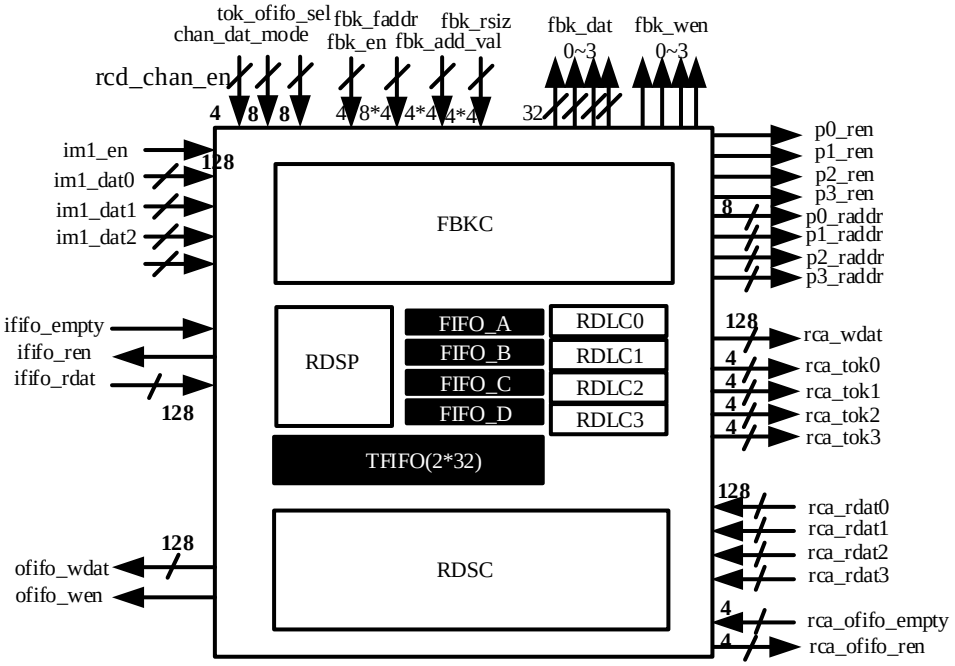
功能说明

2.3.5 RCA Data Controller

结构框图



方案一

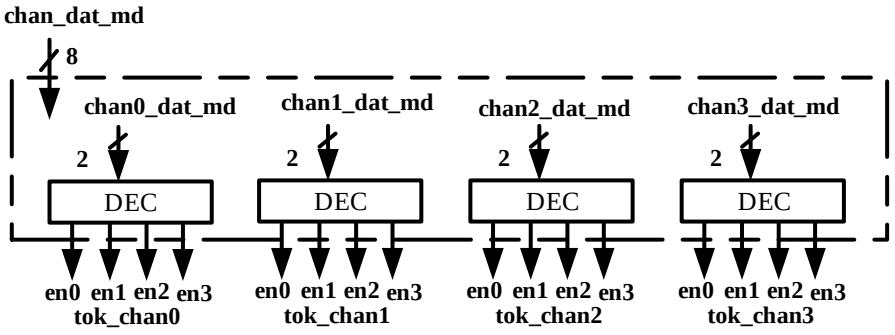


方案二

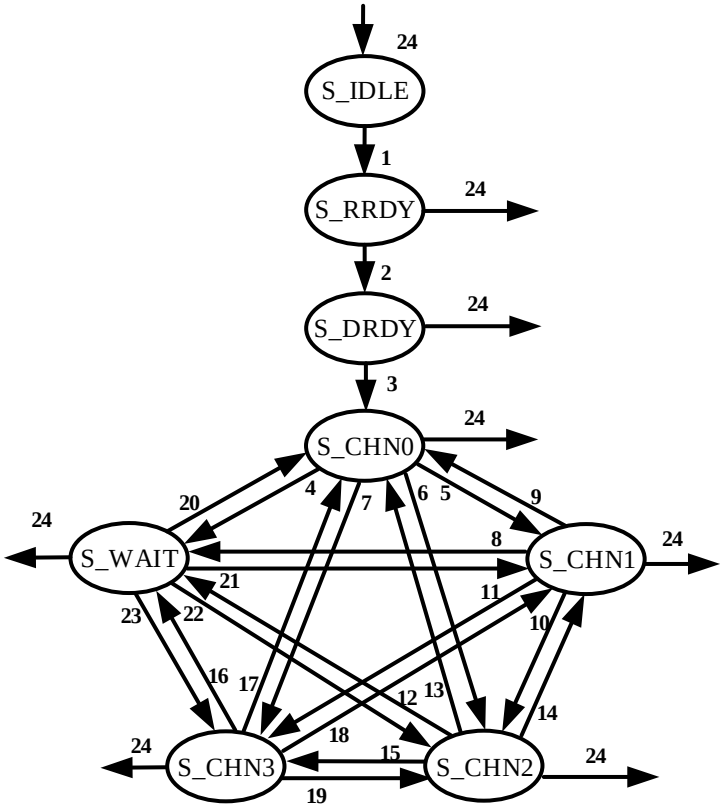
功能说明：

该模块主要实现数据通路，将输入 FIFO 的数据送入 CMPT 计算模块，并将 CMPT 运算结果送至输出 FIFO，如果 CMPT 需要反馈，同时将反馈数据从 GPRF 中读出，送回参数解析模块。

2.3.5.1 RCA Data Load Controller



方案一状态机图：



功能说明：

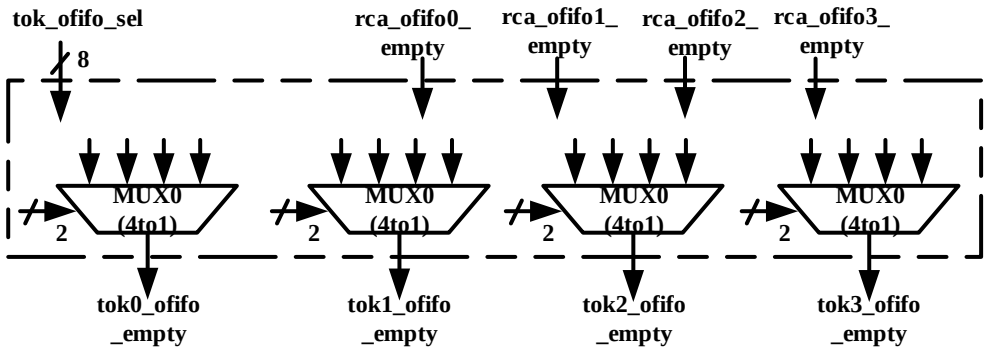
该模块主要将输入 FIFO 的数据，送入 RCA 的 4 个通道，进行处理(RCA 的 4 个通道是有优先级的，通道 0 优先级最高，通道 3 优先级最低)。

条件说明：

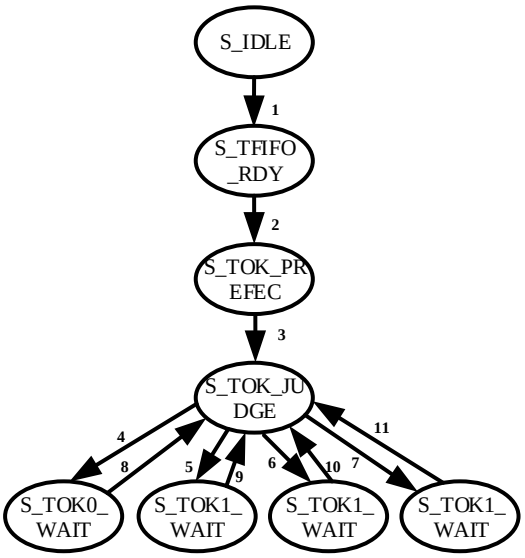
1. RCA 阵列配置完成(rca_cfg_rdy 有效)；
2. 输入 FIFO 非空，数据准备完成；
3. 由于阵列是刚开始工作，4 个通道都是空闲的，通道 0 的优先级最高；
4. 在通道 0 传输数据的状态中：数据有相关性，或者 4 个通道都不能使用(即(chan_en0 & (~rca_lock0)) | (chan_en1 & (~rca_lock1)) | (chan_en2 & (~rca_lock2) | (chan_en3 & (~rca_lock3))的值为低)；
5. 在通道 0 传输数据的状态中：通道 0 不能使用，而通道 1 可以使用；
6. 在通道 0 传输数据的状态中：通道 0 和通道 1 都不能使用，而通道 2 可以使用；
7. 在通道 0 传输数据的状态中：通道 0、通道 1 和通道 2 都不能使用，而通道 3 可以使用；

8. 在通道 1 传输数据的状态中：数据有相关性，或者 4 个通道都不能使用(即(chan_en0 & (~rca_lock0)) | (chan_en1 & (~rca_lock1)) | (chan_en2 & (~rca_lock2) | (chan_en3 & (~rca_lock3))的值为低)；
9. 在通道 1 传输数据的状态中：通道 0 被释放，可以使用；
10. 在通道 1 传输数据的状态中：通道 0 和通道 1 都不能使用，而通道 2 可以使用；
11. 在通道 1 传输数据的状态中：通道 0、通道 1 和通道 2 都不能使用，而通道 3 可以使用；
12. 在通道 2 传输数据的状态中：数据有相关性，或者 4 个通道都不能使用(即(chan_en0 & (~rca_lock0)) | (chan_en1 & (~rca_lock1)) | (chan_en2 & (~rca_lock2) | (chan_en3 & (~rca_lock3))的值为低)；
13. 在通道 2 传输数据的状态中：通道 0 被释放，可以使用；
14. 在通道 2 传输数据的状态中：通道 0 不能使用，而通道 1 被释放可以使用；
15. 在通道 2 传输数据的状态中：通道 0、通道 1 和通道 2 都不能使用，而通道 3 可以使用；
16. 在通道 3 传输数据的状态中：数据有相关性，或者 4 个通道都不能使用(即(chan_en0 & (~rca_lock0)) | (chan_en1 & (~rca_lock1)) | (chan_en2 & (~rca_lock2) | (chan_en3 & (~rca_lock3))的值为低)；
17. 在通道 3 传输数据的状态中：通道 0 被释放，可以使用；
18. 在通道 3 传输数据的状态中：通道 0 不能使用，而通道 1 被释放可以使用；
19. 在通道 3 传输数据的状态中：通道 0、通道 1 都不能使用，而通道 2 被释放可以使用；
20. 在等待状态中：通道 0 被释放，可以使用；
21. 在等待状态中：通道 0 不能使用，而通道 1 被释放可以使用；
22. 在等待状态中：通道 0、通道 1 都不能使用，而通道 2 被释放可以使用；
23. 在等待状态中：通道 0、通道 1 和通道 2 都不能使用，而通道 3 被释放可以使用；
24. 发生复位或者 RCA 阵列整体参数重新加载。

2.3.5.2 RCA Data Store Controller



状态机图：



功能说明：

该模块将 RCA 输出 FIFO 中的数据，按照 RDLC 中送入的先后顺序将结果从 RCA 输出的 4 个 FIFO 中按序取出，写至输出 FIFO。

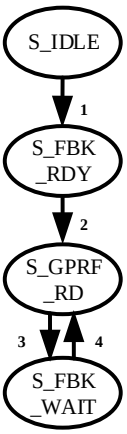
条件说明：

1. TFIFO 中非空；
2. 直接跳转，不需要条件；
3. 直接跳转，不需要条件；
4. 预取的 TOK 值为 0，但 0 对应的输出通道 FIFO 为空；
5. 预取的 TOK 值为 1，但 1 对应的输出通道 FIFO 为空；
6. 预取的 TOK 值为 2，但 2 对应的输出通道 FIFO 为空；

7. 预取的 TOK 值为 3，但 3 对应的输出通道 FIFO 为空；
8. TOK 0 对应的输出通道 FIFO 非空，有数据；
9. TOK 1 对应的输出通道 FIFO 非空，有数据；
10. TOK 2 对应的输出通道 FIFO 非空，有数据；
11. TOK 3 对应的输出通道 FIFO 非空，有数据；

2.3.5.3 Feedback Data Controller

状态机图：



功能说明：

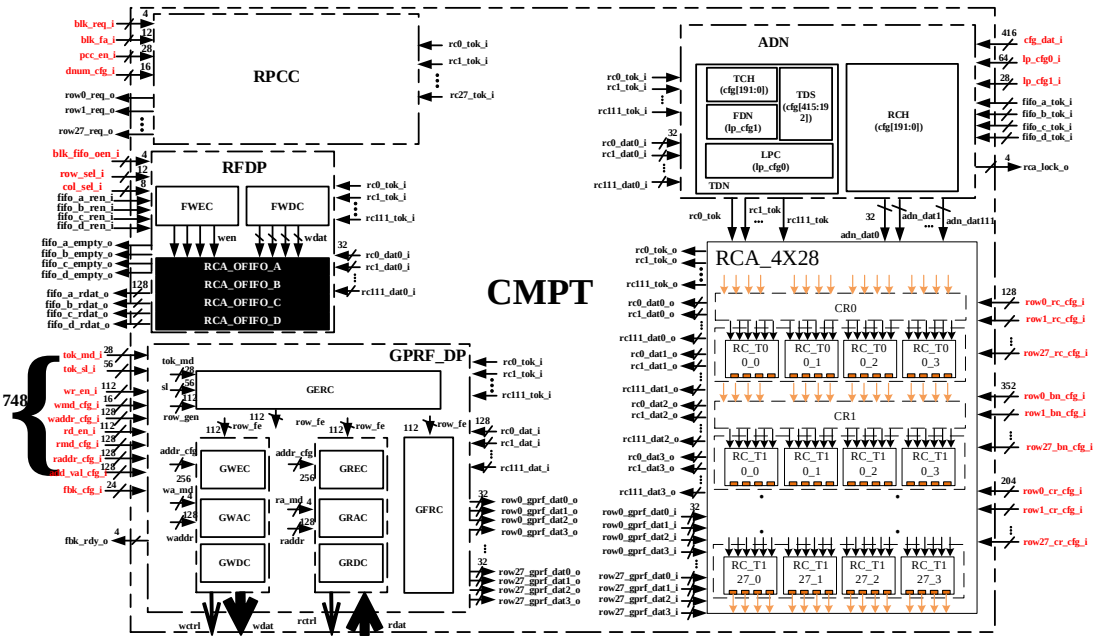
反馈控制模块中有 4 个反馈支路，每个反馈支路的状态机都一样，如上图所示。

条件说明：

1. 反馈路径上 Ready 信号有效；
2. 直接跳转，无需条件；
3. Ready 信号无效；
4. Ready 信号有效。

2.4 Computing Controller

结构框图:



功能说明:

该模块为核心计算单元，接收配置 RCC 送过来的配置参数，和 RCD 送过来的带运算数据，处理后送出。

RCA(Reconfigurable Array)，为可重构阵列，主要实现数据的运算；

ADN(Asynchronous Drive Network)，为异步驱动使能网络，对可重构阵列 TOK 驱动使能网络进行管理，ADN 设计主要用于提供数据在 RCA 阵列中跨行或者跨列(即阵列中任意 2 个位置的 RC 有数据交互)的跳转传输的方式。

GPRF_DP(GPRF Datapath)，为通用寄存器数据通路模块，实现可重构阵列对通用寄存器的读写操作管理，GPRF 设计主要用于存放 RCA 阵列短期内不需要使用的数据，或者当 ADN 中 TCH(RCH)不够使用时，提供另外一种跳转传输方式，但是性能不如 ADN。

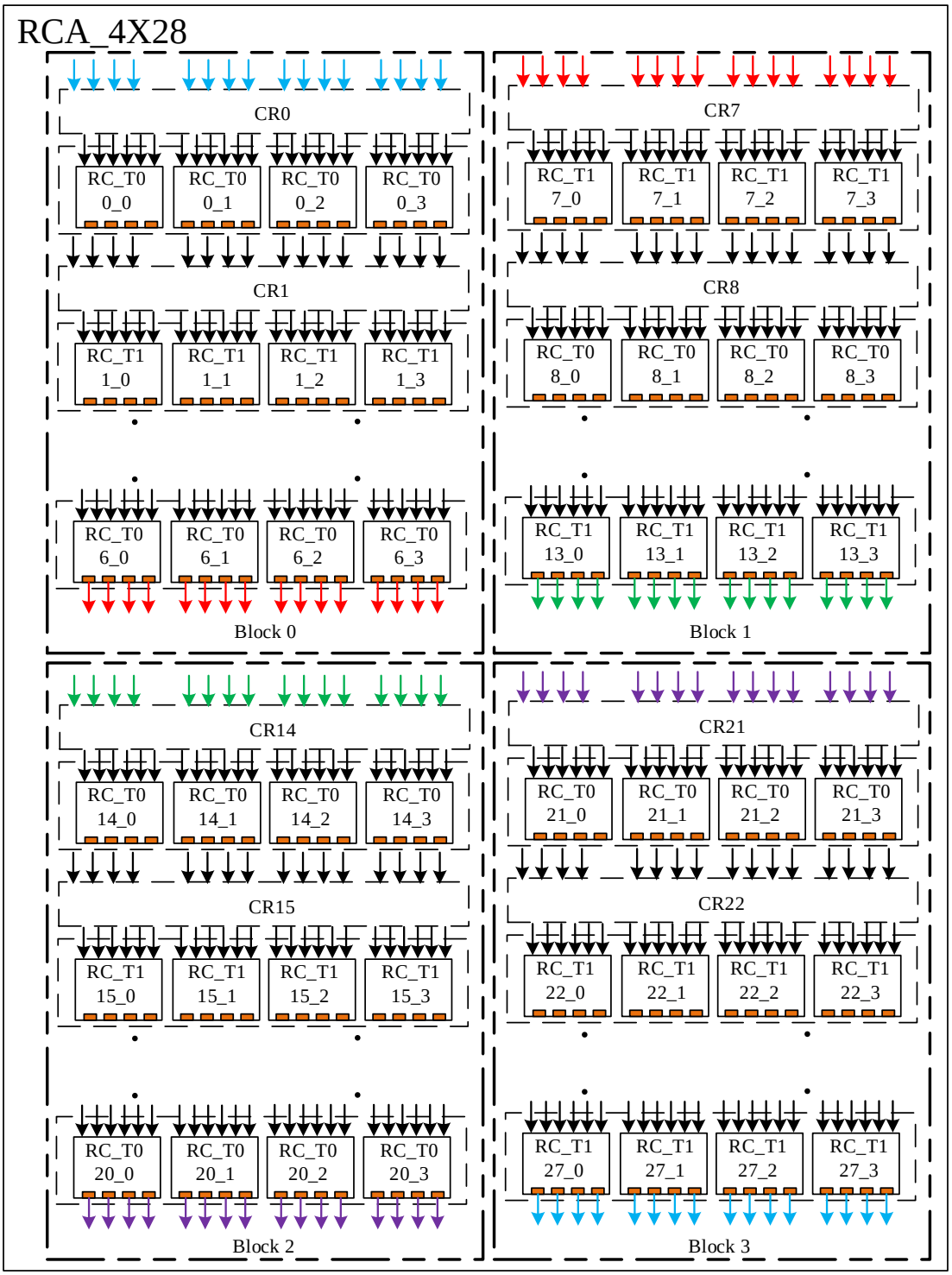
RPCC(RCA Parameter Change Controller)，为可重构阵列参数自更换请求控制器。

RFDP(RCA FIFO Datapath)，为可重构阵列 FIFO 数据通路。

需要的配置参数有 RPCC(60bit)，RFDP(24bit)，GPRF_DP(748bit)，ADN(508bit)，RCA(3584+4928+5712+X(SBOX)即 14224bit)，共计最大为 15564bit+X(SBOX)的配置参数。

2.4.1 Reconfigurable Array

结构框图：



功能说明：

该模块为可重构阵列，阵列大小为 4*28（行编号为 0~27），阵列中共有 2 种 RC 单元(RC_T0 和 RC_T1)。RCA 阵列中共有 14 行(4 个 RC 为 1 行)Type 0 类

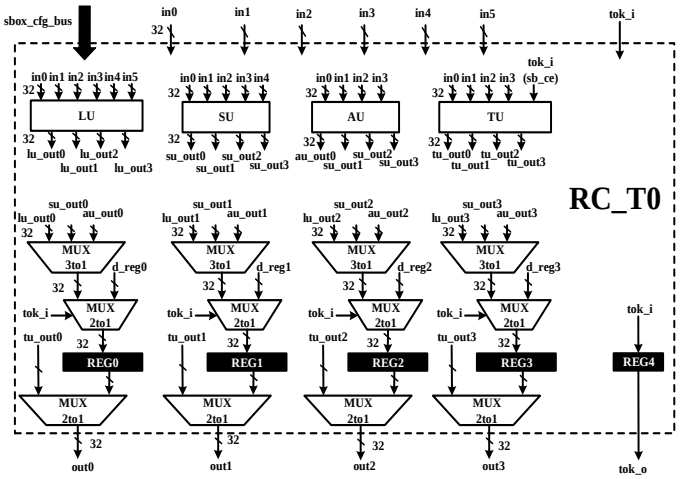
型的 RC，和 14 行 Type 1 类型的 RC。

数据在阵列中正常流动方式为从上向下，同时在最后 1 行（第 27 行）可以返回到第 0 行。

同时为了简化数据路由网络规模，将阵列分成 4 个块，即每个块的大小为 4 行。第 0 到第 6 行的 RC 阵列划分为 Block 0；第 0 到第 6 行的 RC 阵列划分为 Block 1；第 0 到第 6 行的 RC 阵列划分为 Block 2；第 0 到第 6 行的 RC 阵列划分为 Block 3。

2.4.1.1 Reconfigurable Cell Type0

结构框图：



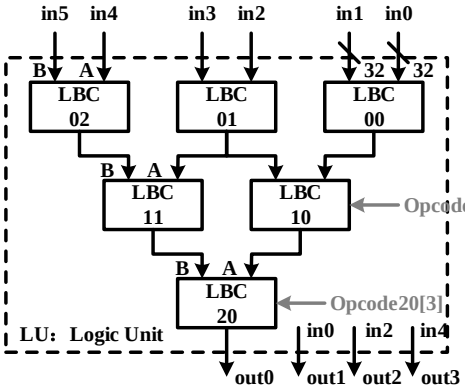
ALU.CFG	ALU
2’b00	LU
2’b01	SU
2’b10	AU
2’b11	TU

功能说明：

RC_T0 为 RCA 阵列的基本单元，包含的 ALU 运算单元有 LU(Logic Unit)，AU(Arithmetic Unit)，SU(Shift Unit)，TU(Lookup Table Unit)。下面将分别对这些运算单元说明。

LU(Logic Unit)：

结构框图：



CFG.LU[18]:
{Opcode20,Opcode11,
Opcode10,Opcode02,
Opcode01,Opcode00}

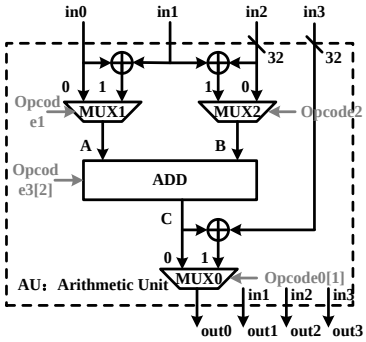
OpcodeXX	算子
3’b000	A
3’b001	~A
3’b010	A^B
3’b011	(~A)^B
3’b100	A&B
3’b101	(~A)&B
3’b110	A|B
3’b111	(~A)|B

功能说明：

主要实现逻辑运算功能，比如与、或、异或等，输入 6 个操作数。内部由 6 个 LBC(Logic Basic Cell)组成，每个 LBC 需要 3bit 配置参数(配置具体含义如上表所示), 因此 LU 需要 18bit 的配置参数。

AU(Arithmetic Unit)：

结构框图：



CFG.AU[5]:
{Opcode3[2],Opcode2,
Opcode1,Opcode0}

AU.CFG	算子
2’b00	A add8 B
2’b01	A add16 B
2’b10	A add32 B
2’b11	(A+B)mod 2^31-1

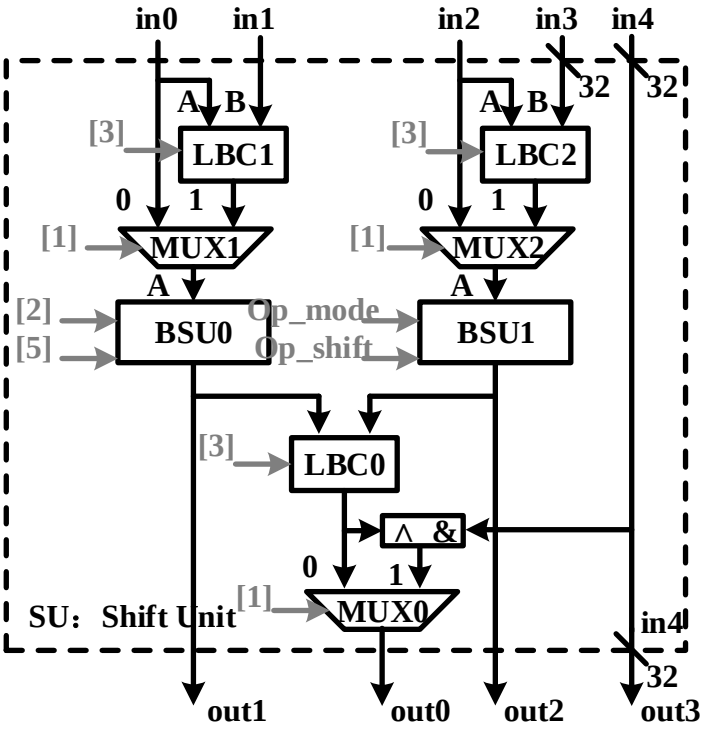
功能说明：

主要实现算术加运算功能，输入 4 个操作数，需要 3bit 配置参数。

SU(Shift Unit)：

结构框图：

Op.modeX	算子
2’b00	A<<op_shift
2’b01	A<<<op_shift
2’b10	A>>op_shift
2’b11	A>>>op_shift



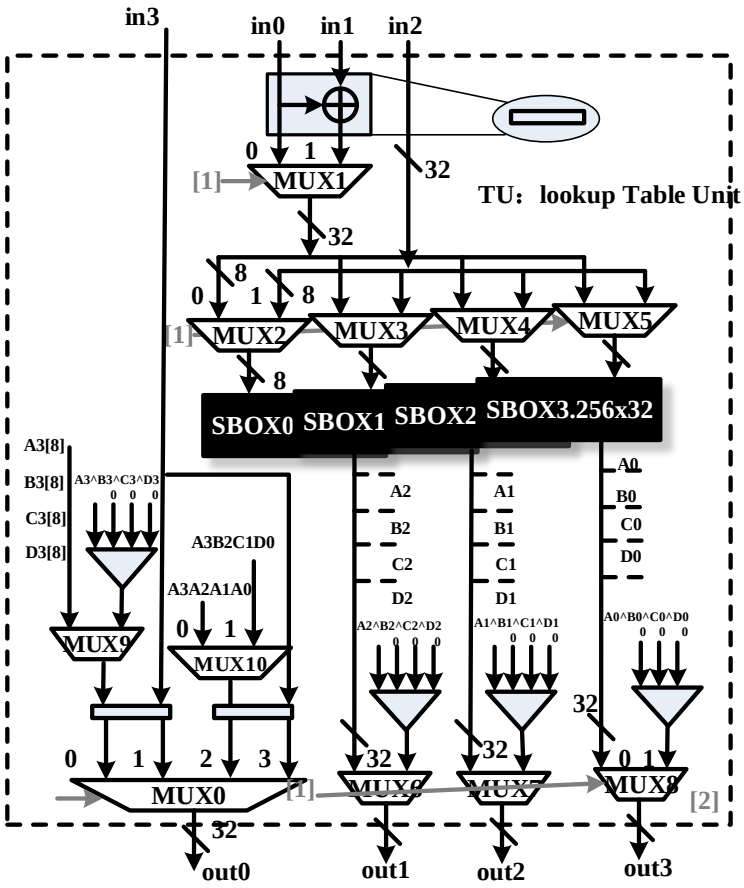
CFG.SU[27]:
{Op.LOGIC,Op.LBC2[3],Op.MUX2,Op.mode1[2],Op_shift1[5];
Op.LBC1[3],Op.MUX1,Op.mode0[2],Op_shift0[5];
Op.LBC0[3],Op.MUX0}

功能说明：

主要实现移位运算功能，输入 5 个操作数，可同时实现 2 个数据的移位运算。需要 26bit 配置参数。

TU(Table Unit)：

结构框图：

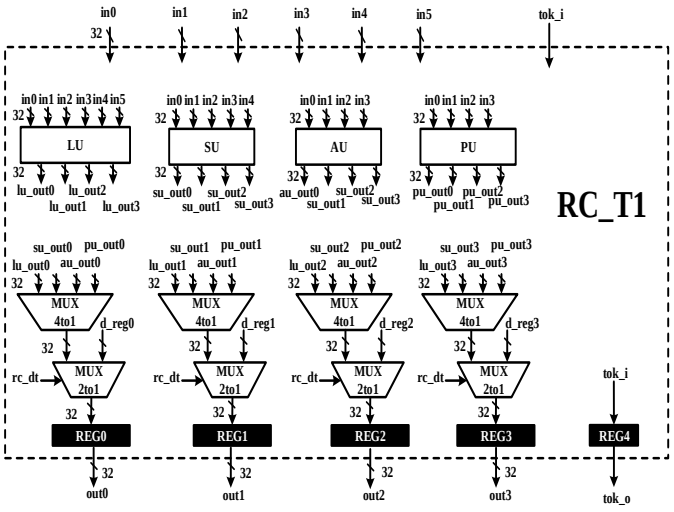


功能说明：

主要实现移位运算功能，输入 4 个操作数，可同时实现 4 个查找表运算。需要 5+X bit 配置参数(X 为 SBOX 的配置参数)。

2.4.1.2 Reconfigurable Cell Type1

结构框图：



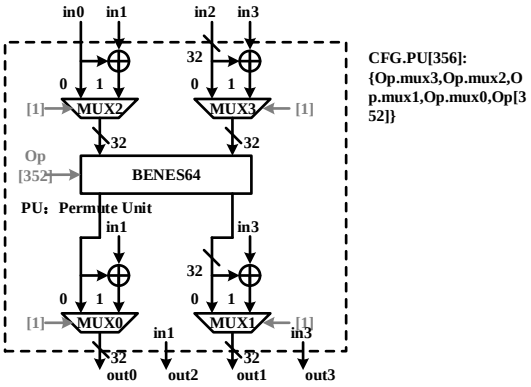
ALU.CFG	ALU
2’b00	LU
2’b01	SU
2’b10	AU
2’b11	PU

功能说明：

RC_T1 包含的 ALU 运算单元有 LU(逻辑运算单元)，AU(加法运算单元)，SU(移位运算单元)，PU(比特 BENES 运算单元)。RC_T1 和 RC_T0 的区别是将 TU 更换成 PU。其中 LU、AU、SU 和 TC_T0 中一样，在上一节有描述，这一节只描述 PU。

PU(Permutation Unit)：

结构框图：



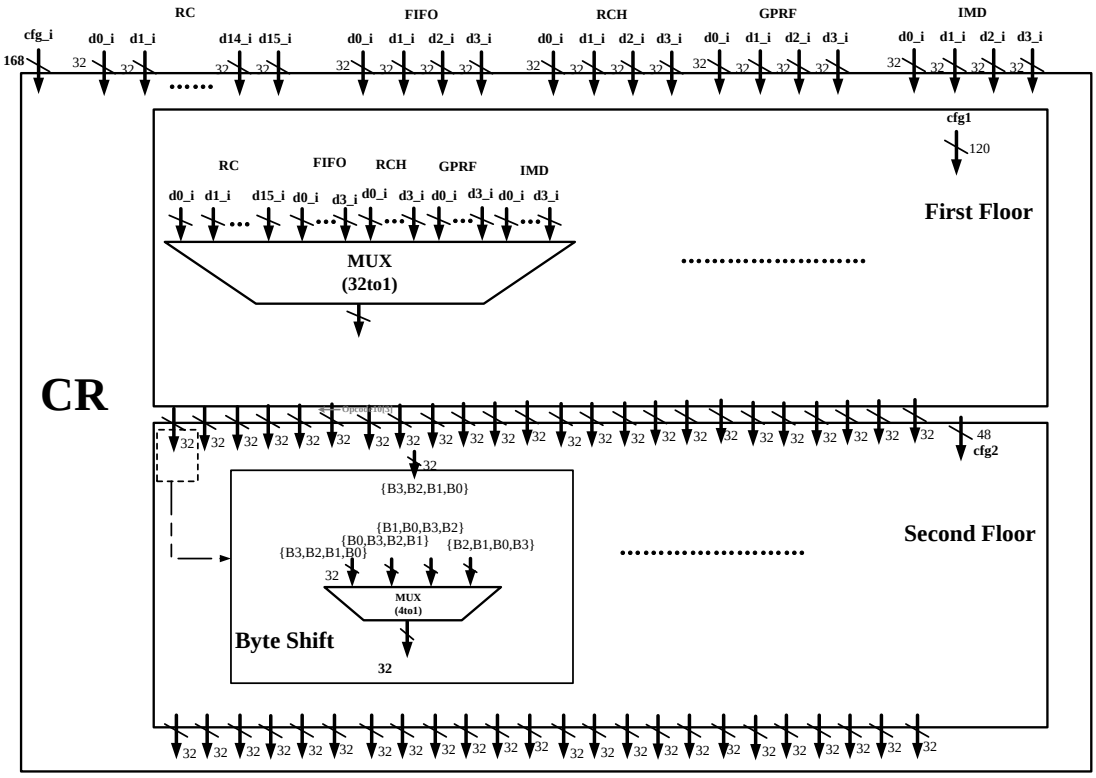
功能说明：

PU 是置换控制单元，由一个 64 位的 BENES（无阻塞）网络和 4 组异或逻辑构成。需要 4+352bit 配置参数。

BENES (N).length = (n/2)*(2*log2 (n)-1), Ex. 64 位的配置信息为 32*11=352。

2.4.1.3 Connection Row

结构框图：



功能说明：

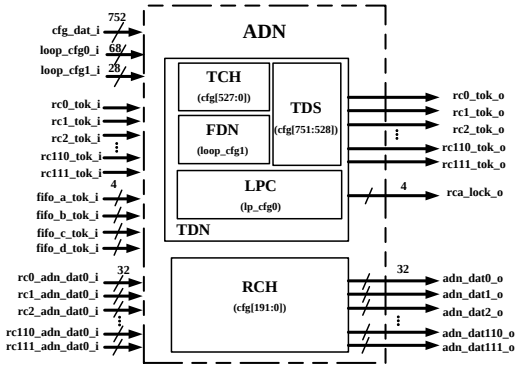
CR 模块主要实现可重构单元行之间的数据路由和选择。由 2 部分组成。

第一层主要实现数据的全互联选择：

第二层实现数据(24 个字)内的字节重组，可以实现一个字内基于字节的各种操作。

2.4.2 Asynchronous Drive Network

结构框图:



功能说明:

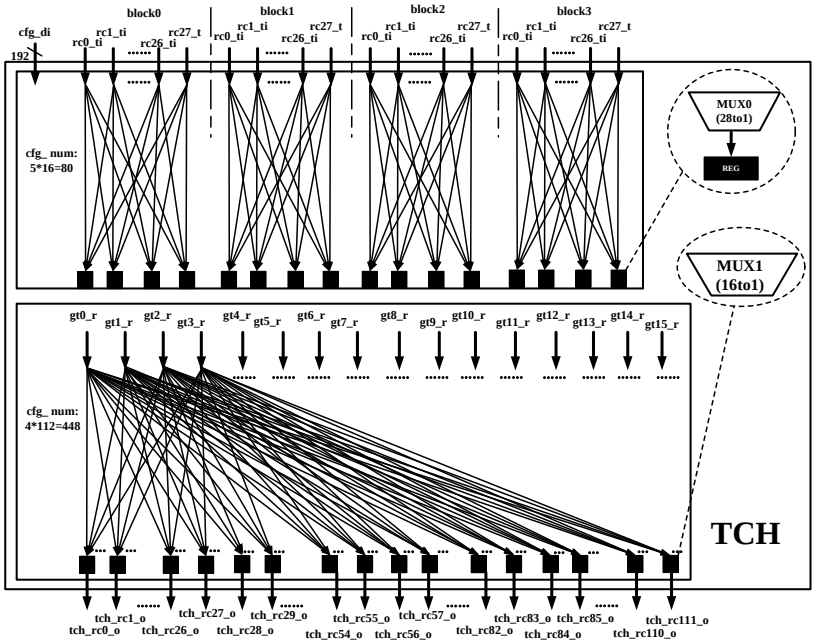
ADN 模块为 RCA 阵列的异步令 TOK 使能驱动网络，通过驱动使能的流水

实现整个阵列的数据计算流动。ADN 内包括 TDN(TCH、FDN、TDS、LPC)和 RCH。

LPC

2.4.2.1 Tok Drive Network

TCH(Tok Channel)结构框图：



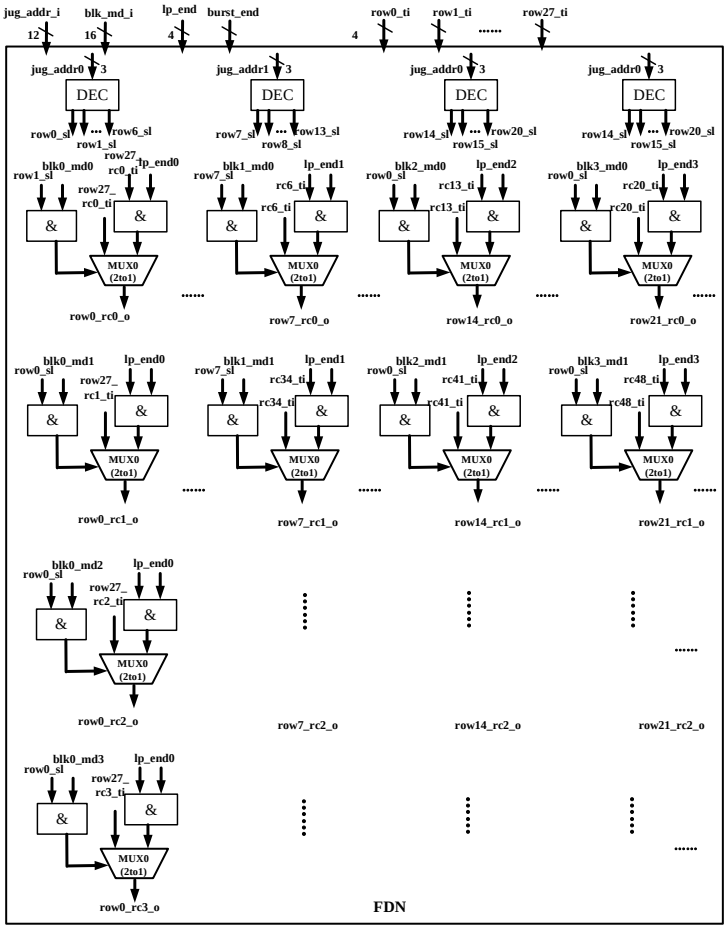
功能说明:

TCH 为通用 TOK 通道(内有 16 个子通道)，接收 112 个 RC 单元输出 TOK，经过选择寄存后，再分配到各个 RC。主要功能实现分为二层。

第一层 16 个 TOK 通道分区域接收 112 个 RC 单元的 TOK 输出，其中编号为 0~3 的通道接收 Block 0(共计 28 个 RC)的 TOK 输出；编号为 4~7 的通道接收 Block 1(共计 28 个 RC)的 TOK 输出；编号为 8~11 的通道接收 Block 2(共计 28 个 RC)的 TOK 输出；编号为 12~15 的通道接收 Block 3(共计 28 个 RC)的 TOK 输出。每个区域内的 4 个子通道对本 Block 的 28 个 RC TOK 输出做 28 选 1 的全互联，选择以后再经过寄存器寄存操作。本层实现主要由 16 个 28 选 1 的 MUX 和 16 个寄存器构成。

第二层将 16 个通道的 TOK 对 112 个 RC 实现全互联，

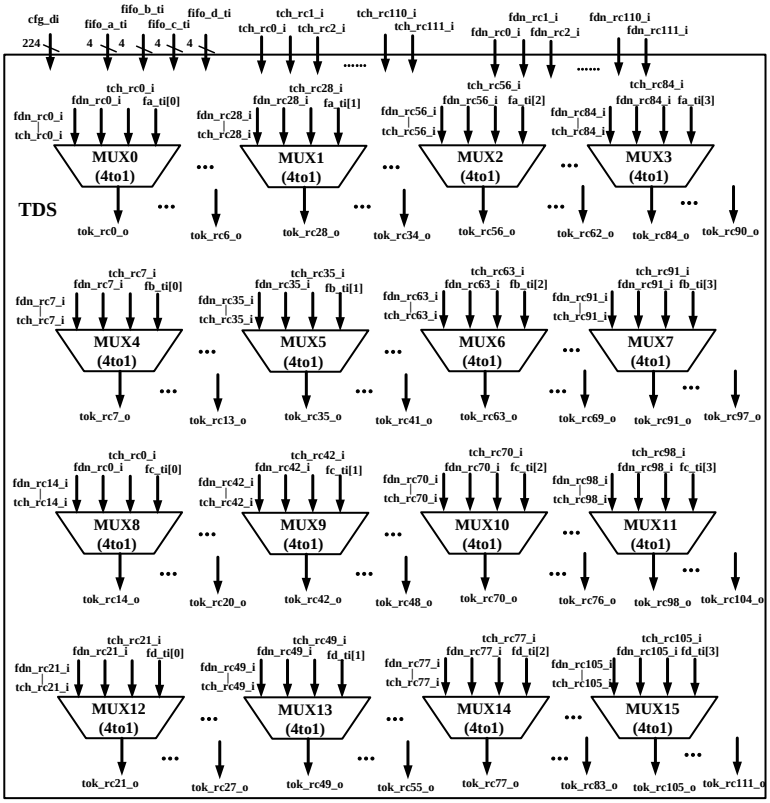
FDN(Fixed Tok Network)结构框图：



功能说明:

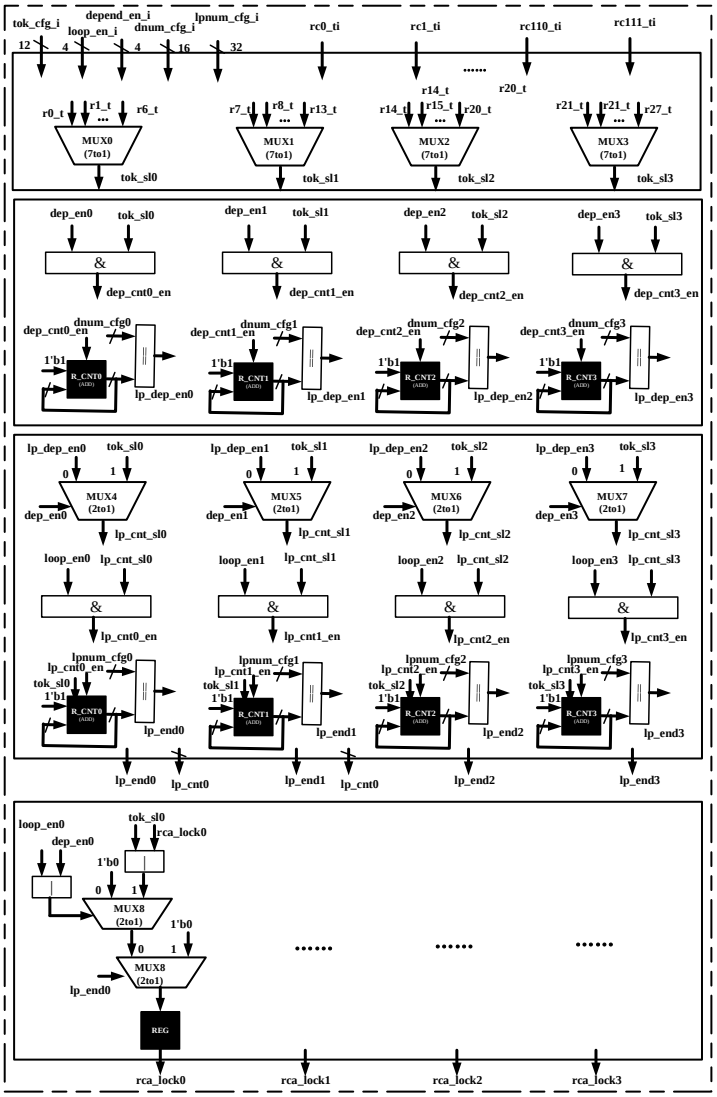
FDN 为固定 TOK 驱动网络，RC 之间的 TOK 连接方式是固定的。

TDS(Fok Drive Select)结构框图：



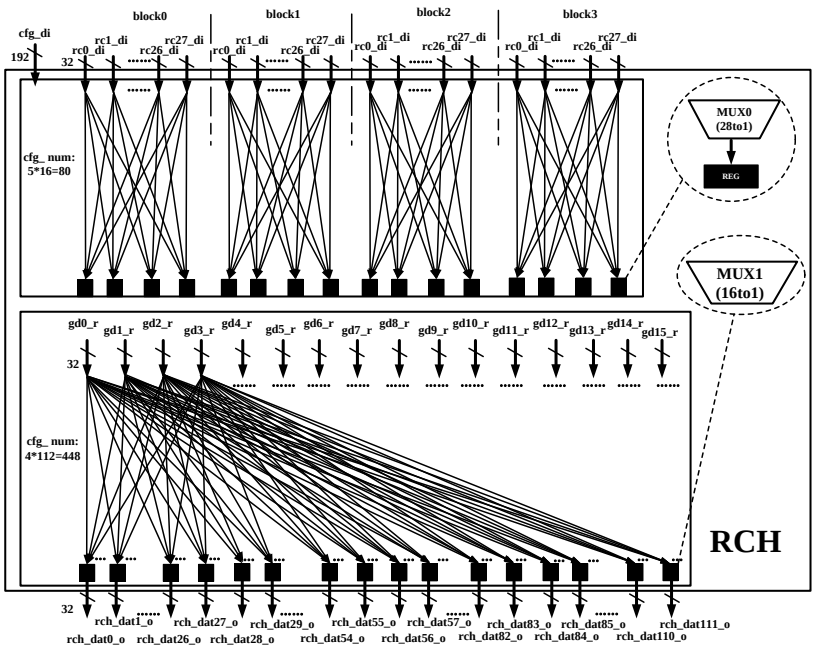
功能说明:

LPC (Loop Controller)结构框图：



2.4.2.2 Registers Channel

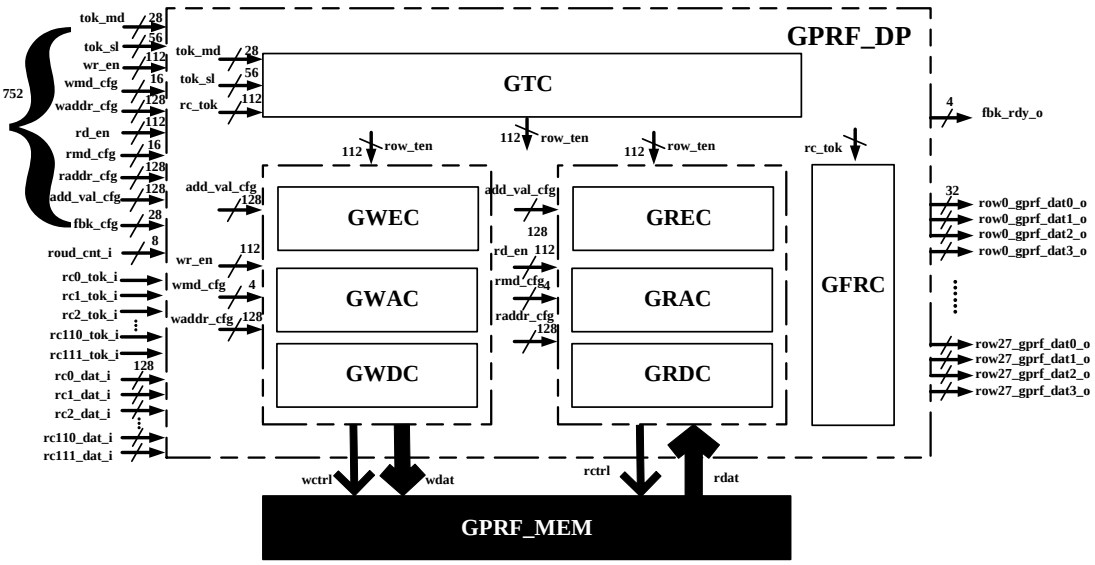
结构框图



功能说明

2.4.3 GPRF Datapath

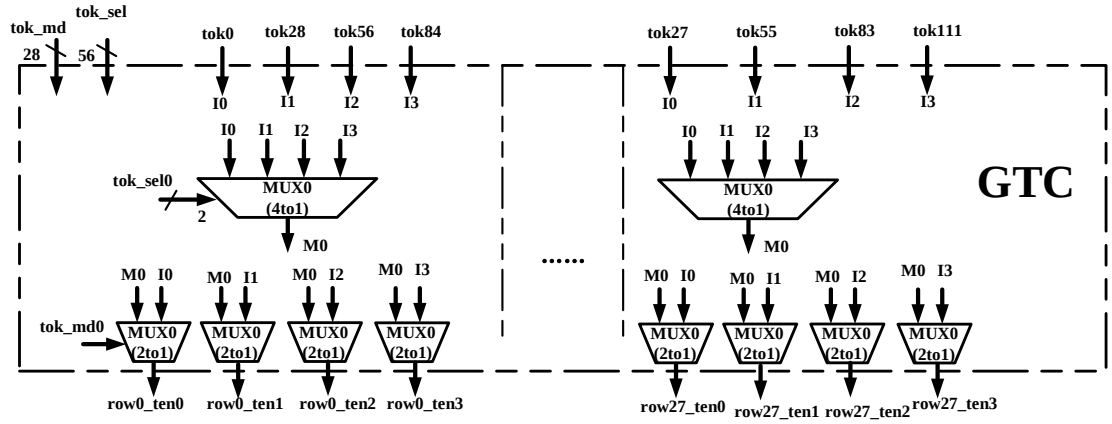
结构框图：



功能说明:

2.4.3.1 GPRF Enable Row Controller

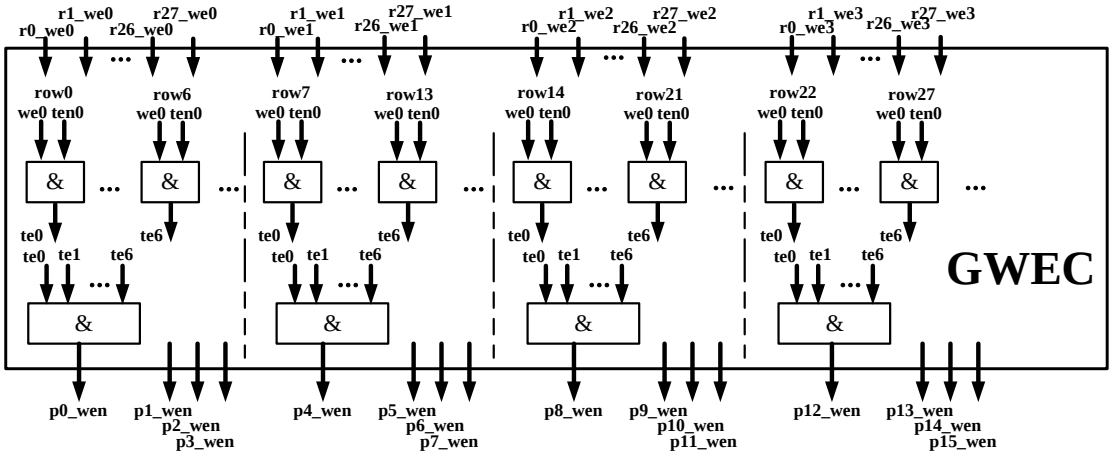
结构框图：



功能说明：

2.4.3.2 GPRF Write Enable Controller

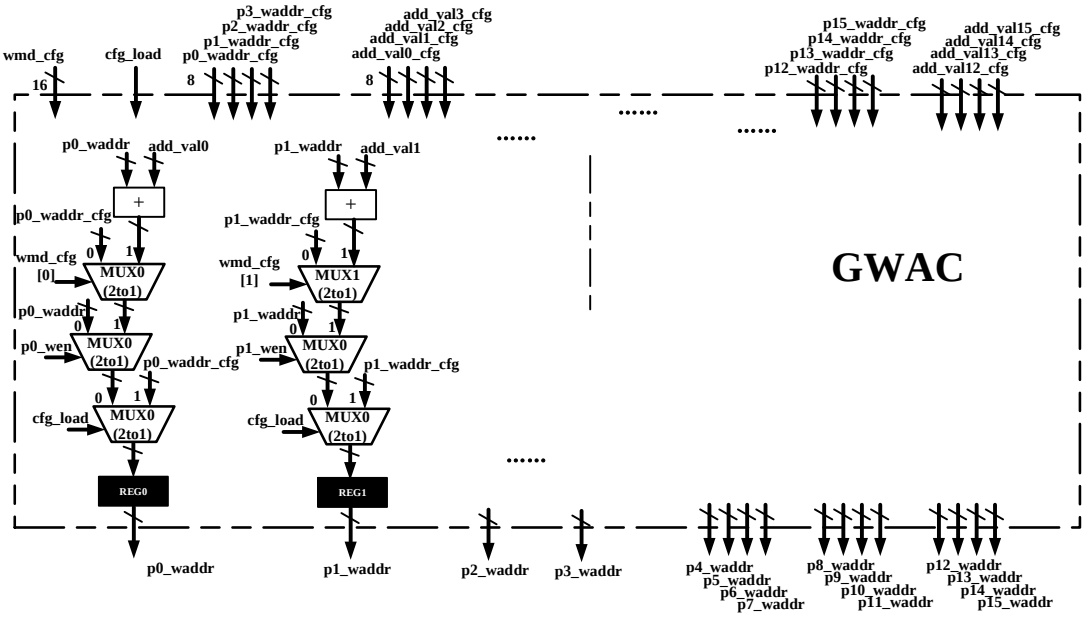
结构框图：



功能说明：

2.4.3.3 GPRF Write address Controller

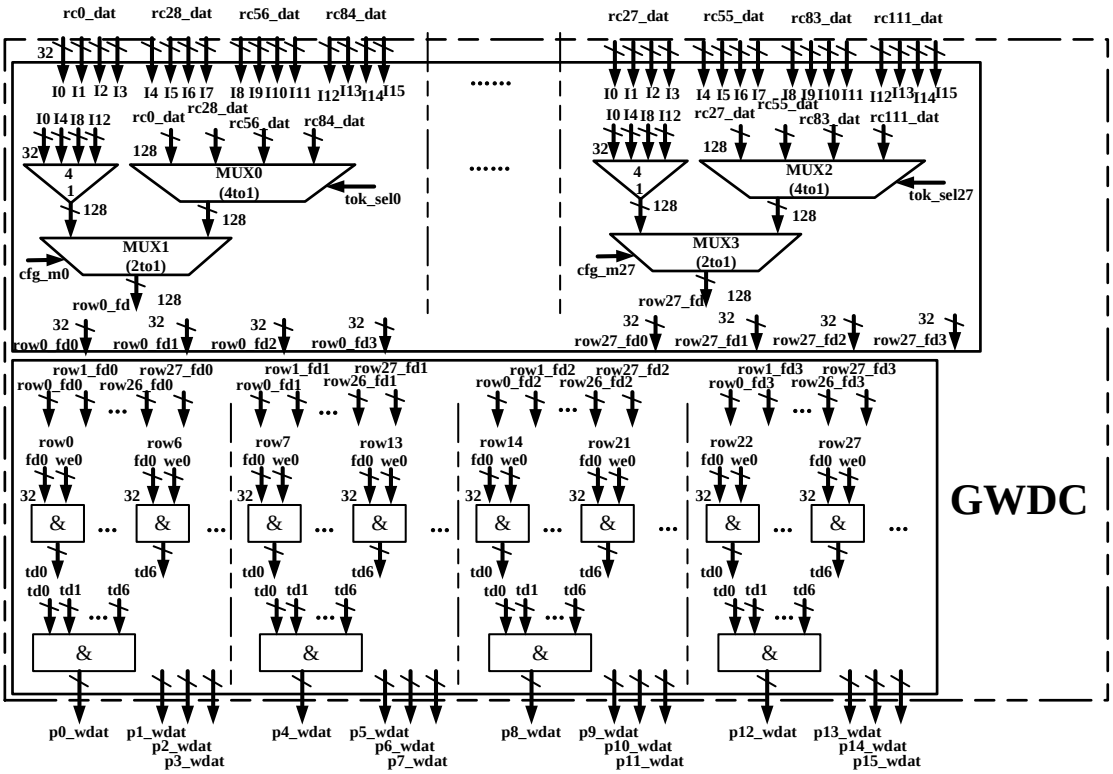
结构框图：



功能说明：

2.4.3.4 GPRF Write Data Controller

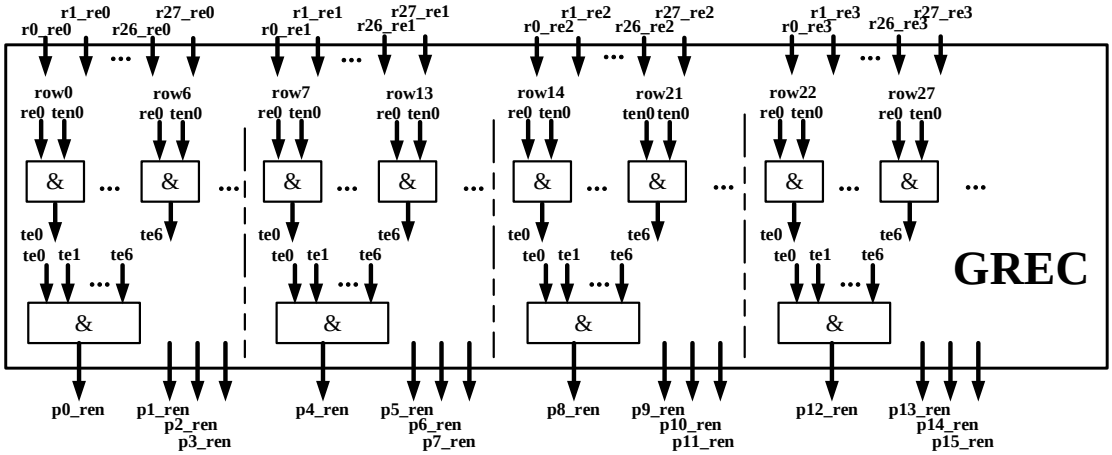
结构框图：



功能说明：

2.4.3.5 GPRF Read Enable Controller

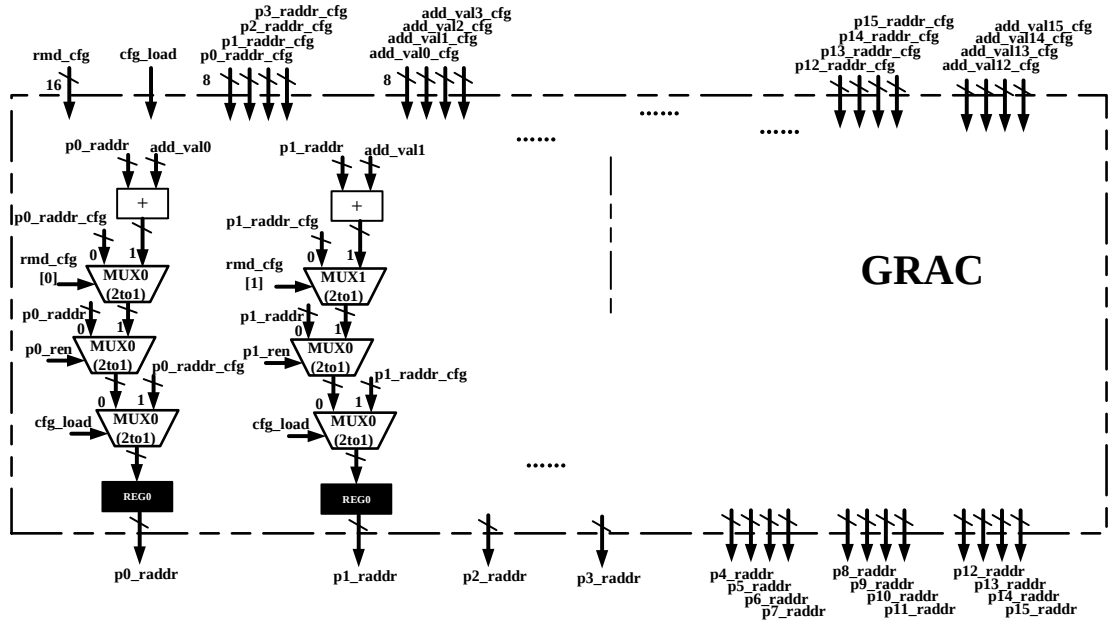
结构框图：



功能说明：

2.4.3.6 GPRF Read Address Controller

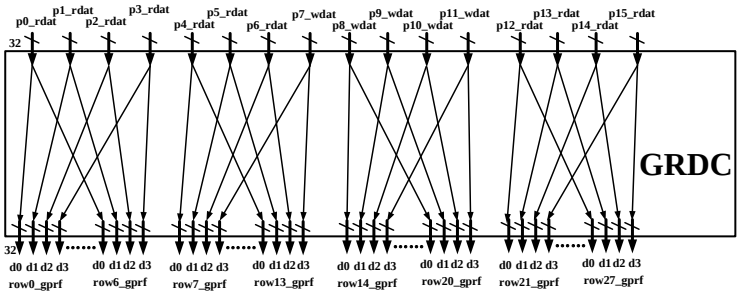
结构框图：



功能说明：

2.4.3.7 GPRF Read Data Controller

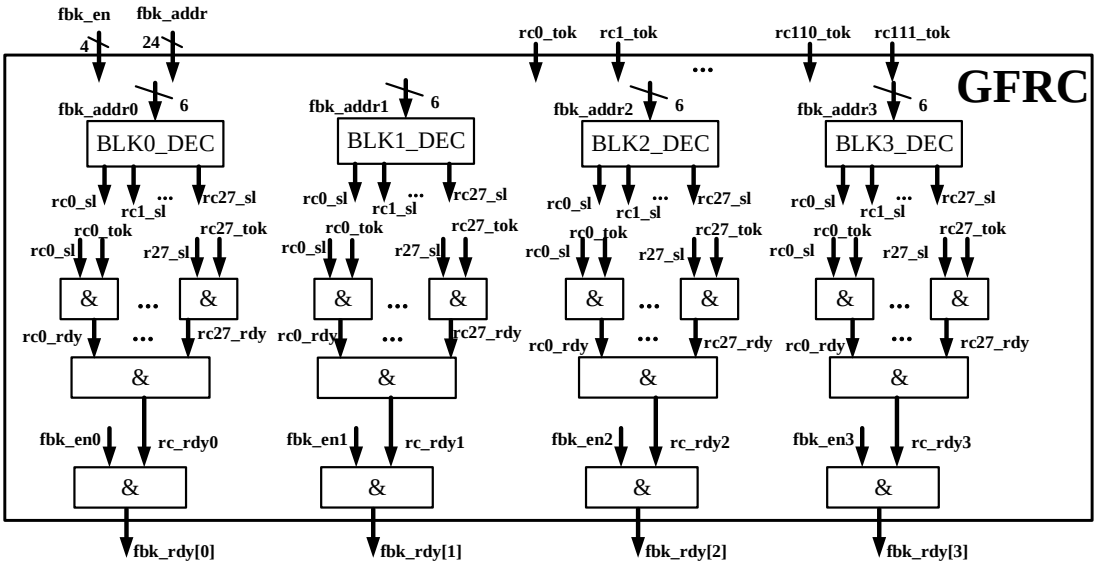
结构框图：



功能说明：

2.4.3.8 GPRF Feedback Ready Controller

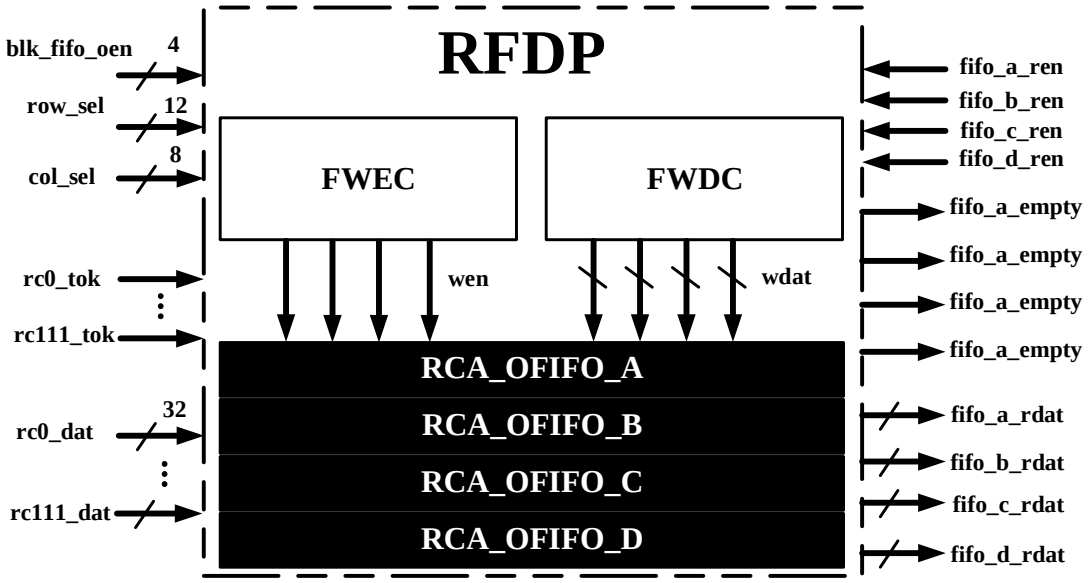
结构框图：



功能说明：

2.4.4 RCA FIFO Datapath

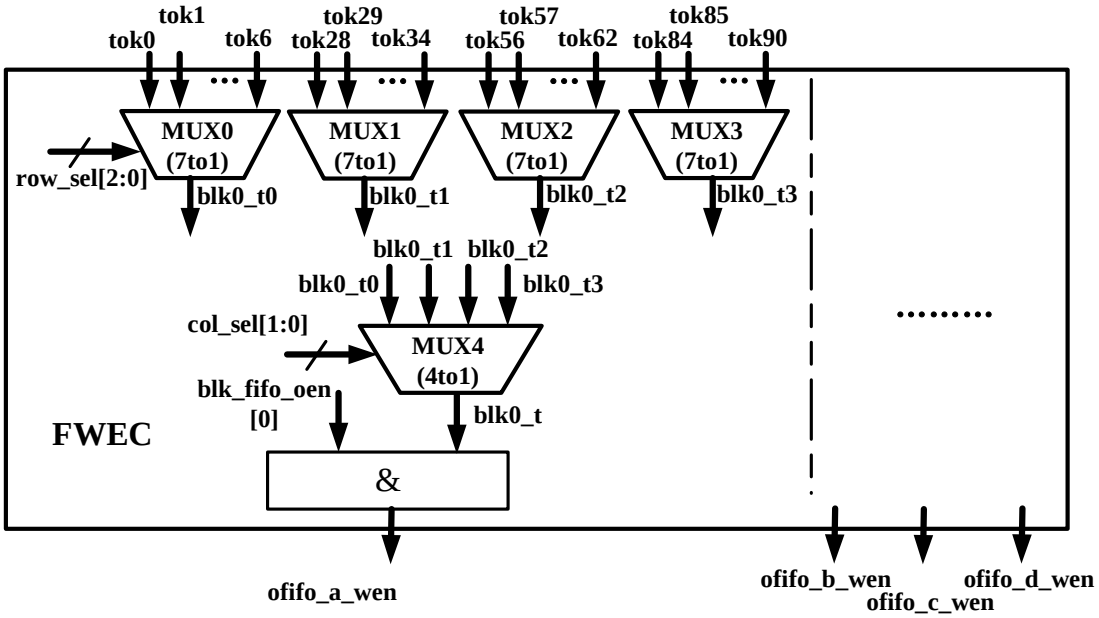
结构框图：



功能说明：

2.4.4.1 FIFO Write Enable Controller

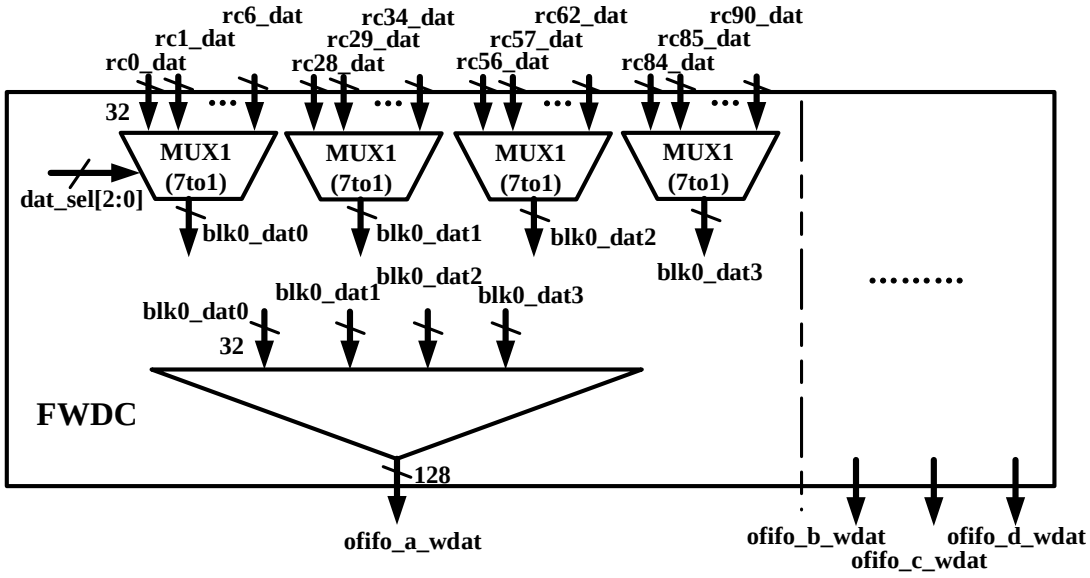
结构框图：



功能说明：

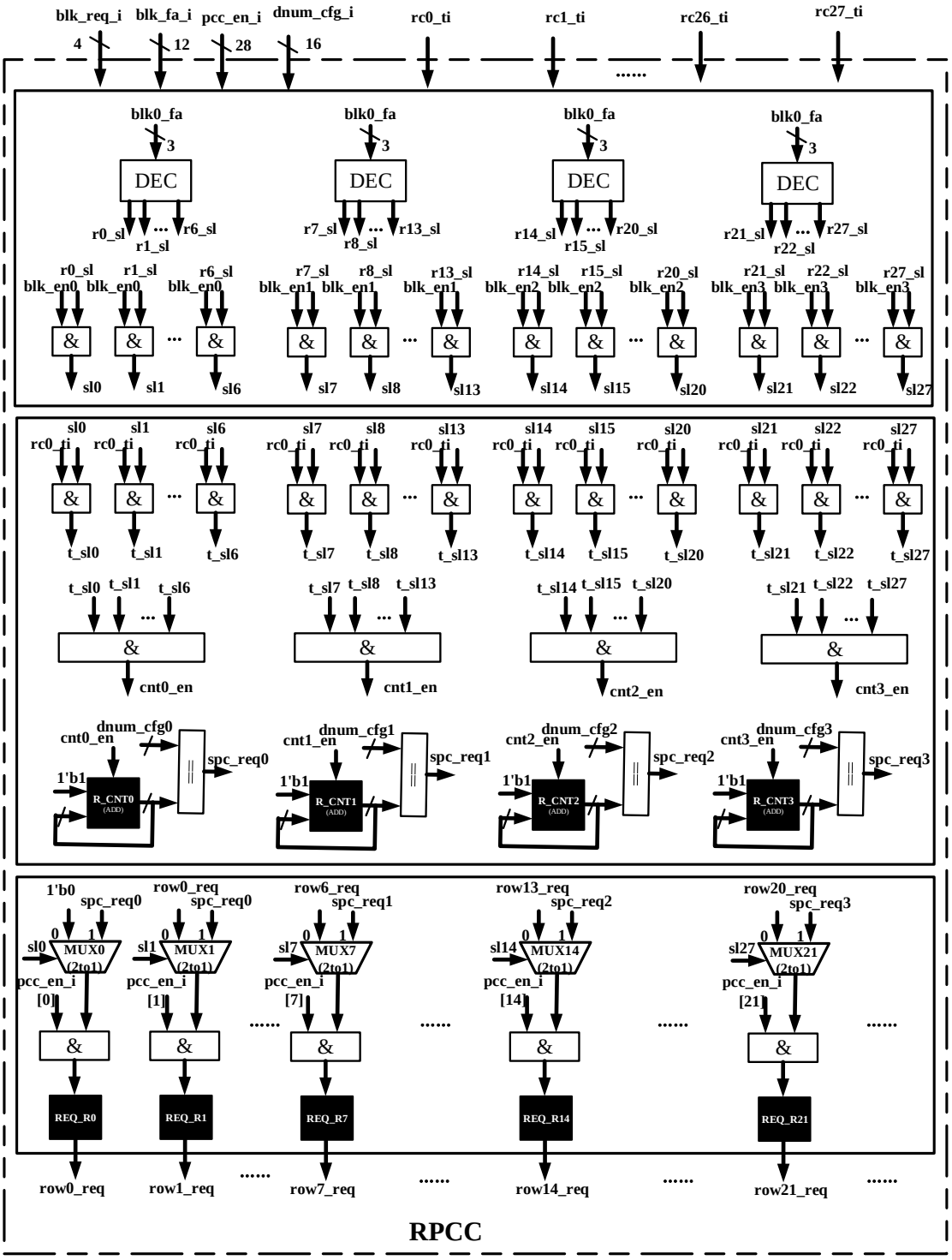
2.4.4.2 FIFO Write Data Controller

结构框图：



2.4.5 RCA Parameter Change Controller

结构框图：



2. 项目文件树结构

文件存放于 mercury 服务器

(0) ----------rpu.v (顶层)

(1) ----------rpu_rcc.v (可重构配置参数控制器)

(2) ----------rcc_cfg_intf.v (配置参数接口)

(2) ----------rcc_ctx0_mem.v (L0 Context 配置参数存储器)
(3) ----------p_rc_ram.v (RC 单元配置参数存储器)
(3) ----------p_cr_ram.v (CR 单元配置参数存储器)
(3) ----------p_pu_ram.v (BENES 配置参数存储器)
(3) ----------p_sb_ram.v (SBOX 配置参数存储器)
(3) ----------p_imd0_ram.v (IMD0 存储器)
(3) ----------p_imd1_ram.v (IMD1 存储器)
(2) ----------rcc_ctx0_pars.v (L0 Context 解析单元)
(2) ----------rcc_ctx1_mem.v (L1 Context 配置参数存储器)
(3) ----------p_pkt_ram.v (Packet 配置参数)
(3) ----------p_adn_ram.v (ADN 参数存储器)
(3) ----------p_gdp_ram.v (GPRF Datapath 参数存储器)
(2) ----------rcc_ctx1_pars.v (L1 Context 解析单元)

(1) ----------rpu_rcd.v (可重构数据控制器)
(2) ----------rcd_fbkc.v (反馈控制器)
(2) ----------rcd_rdlc.v (数据加载控制器)
(2) ----------rcd_rdsc.v (数据存储控制器)

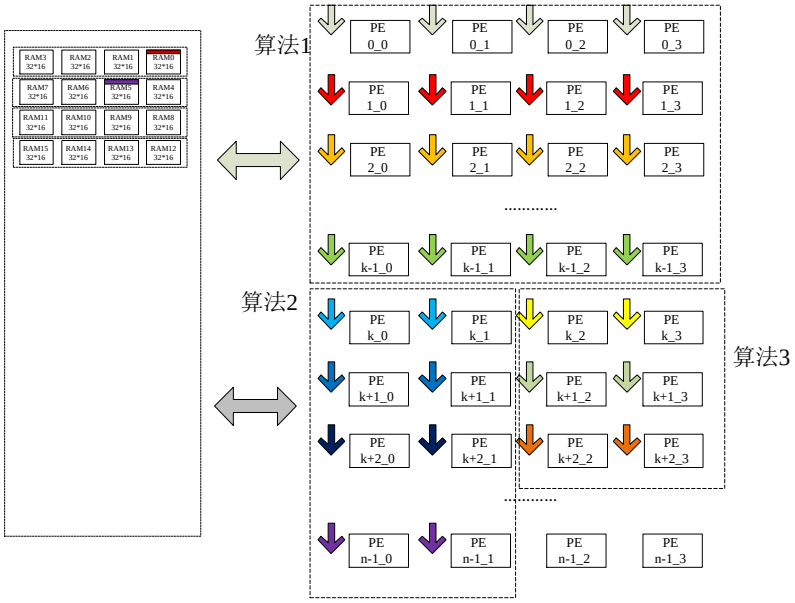
(1) ----------rpu_cmpt.v (可重构计算引擎)
(2) ----------rca.v (可重构阵列)
(2) ----------adn.v ()
(2) ----------gprf_dp.v ()
(2) ----------rpcc.v
(2) ----------rfdp.v

附：流水设计

采用令牌传递流水设计方式，在 PE 阵列设计中，增加一个令牌使能阵列，1 个 PE 单元对应 1 个令牌使能。PE 单元只有接收到对应的令牌使能才开始工作，否则不工作。为此可以将数据在阵列上的流动可以抽象为单 bit 令牌阵列上的令牌流动。

通过令牌流水的方式，可以达到实现 PE 单元的自流水，一定数目的 PE 单独映射，如此在 PE 单元阵列中不但可以映射一套算法，也可以映射多套算法（如下图所示）。甚至由于整个 PE 阵列单元都受令牌使能控制，PE 阵列可以设计成

不带时序信息的异步组合电路，从而达到提高性能和降低功耗的目的。



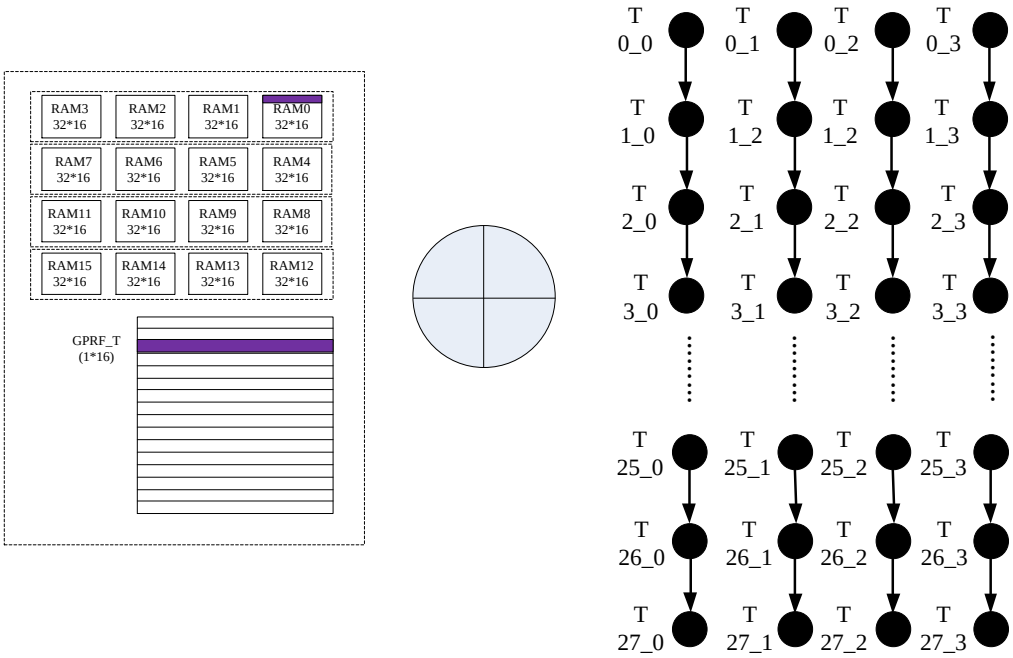
令牌结构设计

特性需求

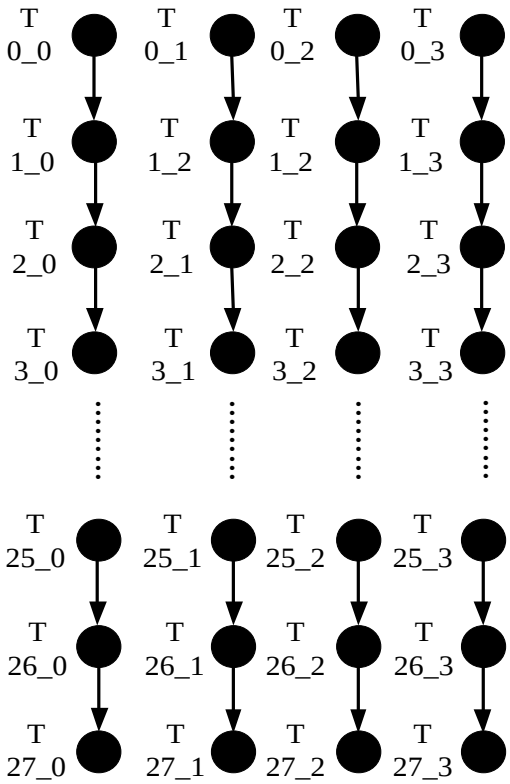
- 上一行的 PE 输出令牌使能被下一行的 PE 使用；
- 有循环，指定行的 PE 接收来自下面第 K 行的 PE 输出令牌使能；
- 1 个 PE 输出令牌使能被多个 PE 单元使用；
- 多个 PE 输出令牌使能被 1 个 PE 单元使用（多个 PE 可以是同一周期送出令牌使能）。

设计框图

令牌使能结构设计如下图所示，采用固定令牌使能阵列加通用令牌使能阵列的组合方式，以实现 PE 阵列的自流水运作。



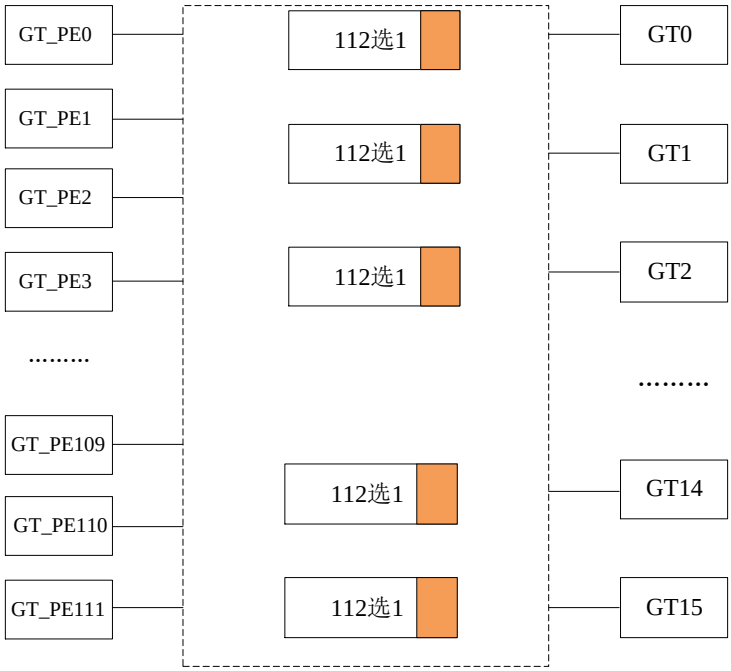
固定令牌使能结构如下图所示，每一个 PE 对应 1 个令牌使能寄存器点，连接方式为固定的，上一行的令牌使能直接送入下一行对应的令牌使能点，例如编号为 0_0 的 PE 输出令牌使能只能送给编号为 1_0 的 PE，同一行间不能传递令牌。



令牌流水固定网络结构

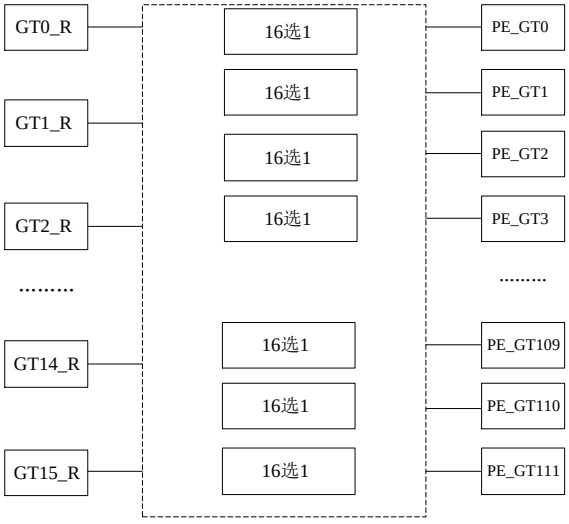
通用令牌使能通道结构如下图所示，接口接收来自 4*28 个 PE 单元输出的

通用令牌使能（112 个），然后送入 16 个通用令牌使能寄存器，最大允许 16 个 PE 单元单独使用通用令牌使能通道。



通用令牌通道 1

16 个通用令牌使能通道输出至 112 个 PE，在每个 PE 内根据配置选择使不使用或者使用那一个通用令牌使能通道送来的令牌使能。



通用令牌通道 2

工作方式

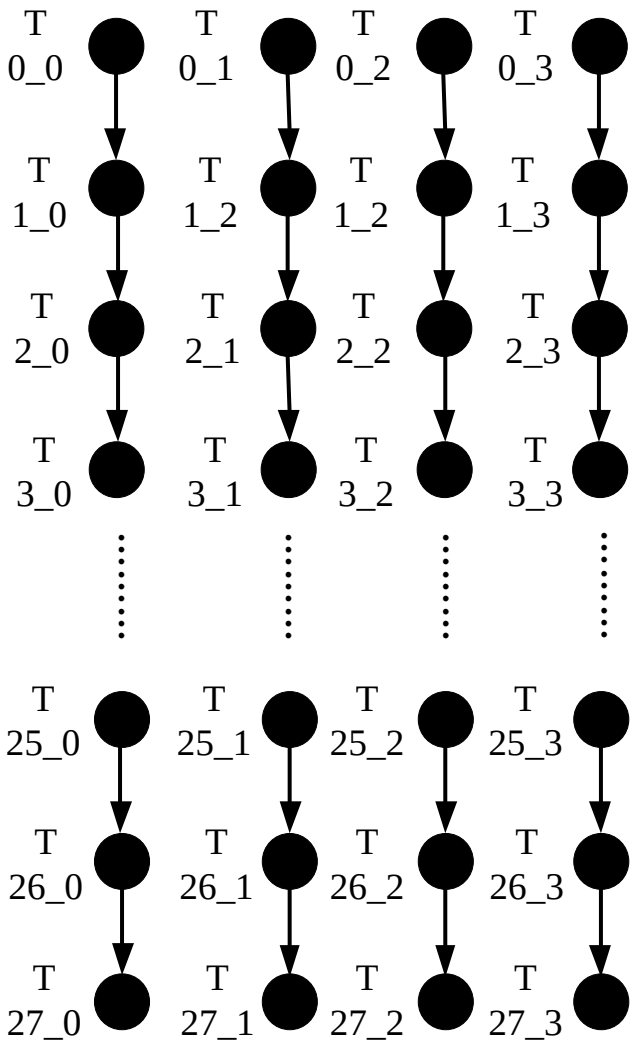
针对顺序结构、分支结构、循环结构的不同模式下的工作方式见 7.2 章节。

理论依据

下面将分别对软件编程中常见的顺序执行、分支执行、循环执行进行讨论，以论证令牌流水的方式能否适应可重构技术中的软硬件双编程。

顺序执行

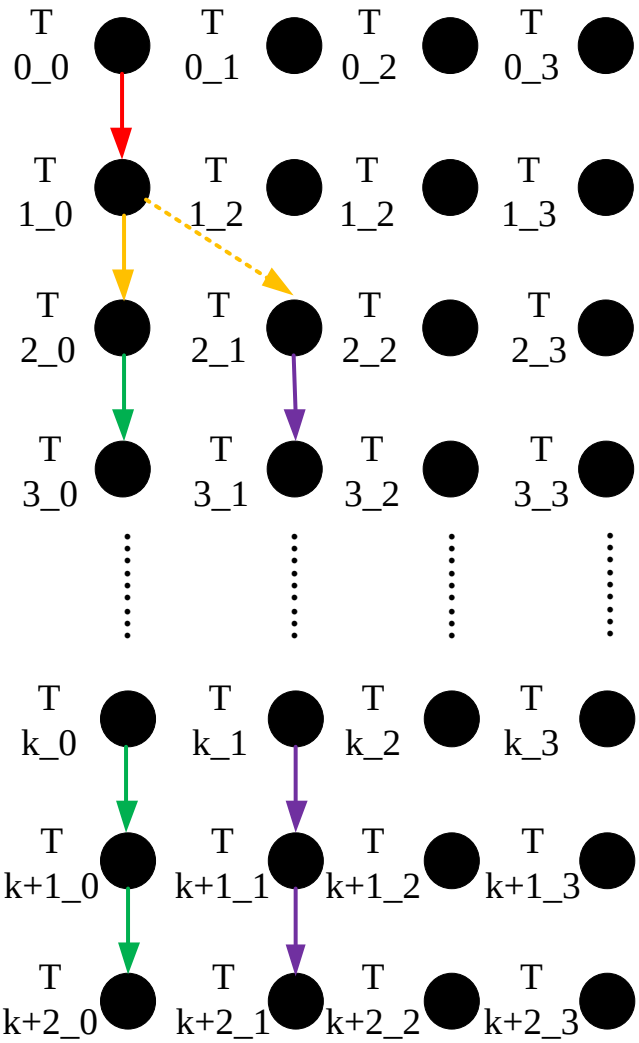
顺序执行属于最常见的执行方式，令牌流映射也相对比较简单，如下图所示。如果 FIFO 非空，且令牌阵列未被锁定，则送出数据和令牌使能给 PE 阵列。



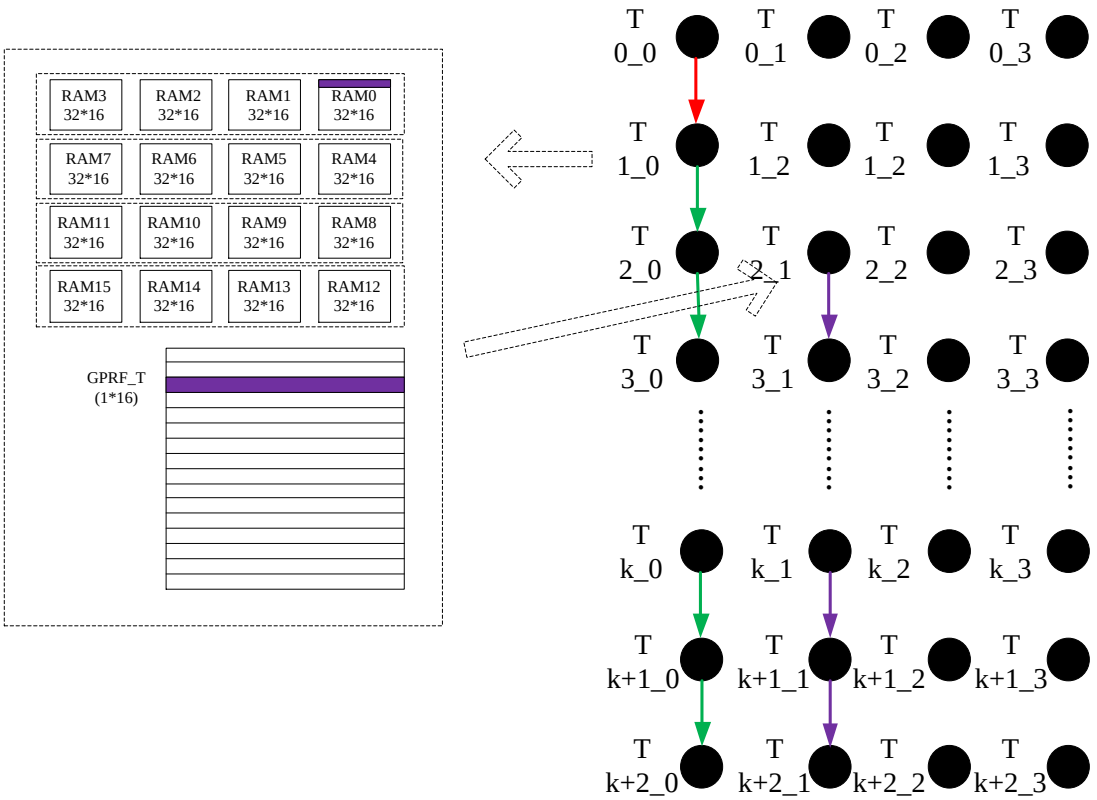
FIFO 将分组数据和令牌送入第一行 PE，该行 PE 处理完以后，将令牌和数据送入下一行 PE，同时接收下一个分组数据和令牌，数据和令牌在 PE 阵列中实现流水。

分支执行

当算法执行过程中出现分支时，如下图所示，PE(编号为 1_0)的输出有 2 种情况（图中绿色和紫色的两条支路）。

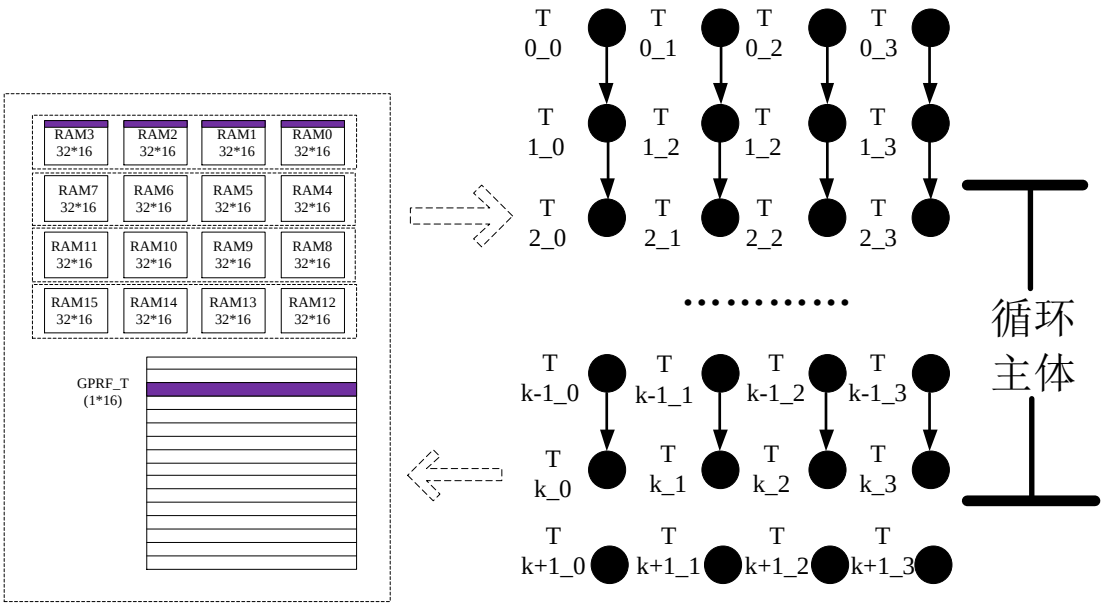


处理方式如下图所示 PE(编号为 1_0)根据运算结果，如果为情况 1，则将令牌使能送给 PE(编号为 2_0)，PE(编号为 2_0)得到令牌使能后，进行运算，执行绿色分支；如果为情况 2，则将令牌送给通用令牌通道，数据存入通用寄存器堆(GPRF)，PE(编号为 2_1)从通用令牌通道取得令牌，然后从通用寄存器堆取数据，进行运算，执行紫色分支。



循环执行

当算法执行过程中出现循环时，如下图所示，PE(行号为第 2 行到第 k 行)需要执行多次循环时，则第 k 行的 PE 运算完成，不能把令牌使能交给第 k+1 行的 PE，而需要把令牌使能交还给第 2 行。



这种情况下，同样需要借用通用令牌通道来实现令牌的传递。第 k 行 PE 执

行完成以后，将令牌送入其中 1 个通用令牌通道（如果是基于行操作，同一行 4 个 PE 都是同一周期工作，则 4 个 PE 可以复用 1 个通用令牌使能通道，否则需要 1 个 PE 使用 1 个通用令牌使能通道），然后第 2 行得到对应通用令牌使能后，开始工作，发出通用寄存器数据读请求，下一个周期后得到对应数据，开始工作。

映射实现

单个 PE 有两种工作方式：一、不需要读取 GPRF，则 PE 工作流程为令牌——写回，如图 7.1 所示，令牌在 PE 内只停留 1 个周期；二、需要读取 GPRF，则 PE 工作流程为令牌——取数——写回，如图 7.2 所示，令牌在 PE 内需要停留 2 个周期。

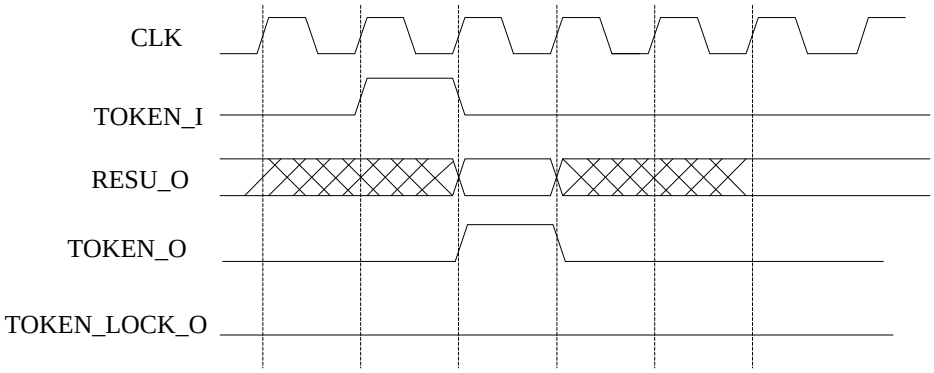


图 7.1　PE 令牌流水方式 1

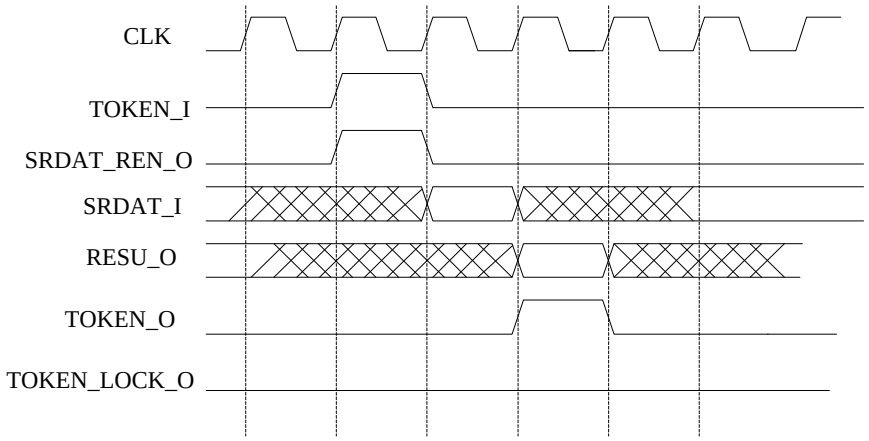


图 7.2　PE 令牌流水方式 2

进入数据处理后，FIFO 判断输入的令牌锁定信号（TOKEN_LOCK）是否有效，如果无效，则送出令牌给 PE，PE 第一个周期接收到令牌后，发出数据读请求；第二个周期得到操作源数据，并进行处理；第三个周期进行运算，并将运算结果写回，同时发出令牌以供下一级的 PE 使用。未接收到令牌的 PE 不工作，

所有数据均为无效状态。

如果算法完全展开，不需要循环，即每个周期都可以处理一组数据，则 PE 只需要按照流程进行数据处理，时序如图 7.1 所示。否则，PE 收到来自 FIFO 的令牌后，将令牌锁定功能使能，时序如图 7.2 所示。由时序图上可以看出，每个令牌在每个 PE 中需要停留 2 个周期。

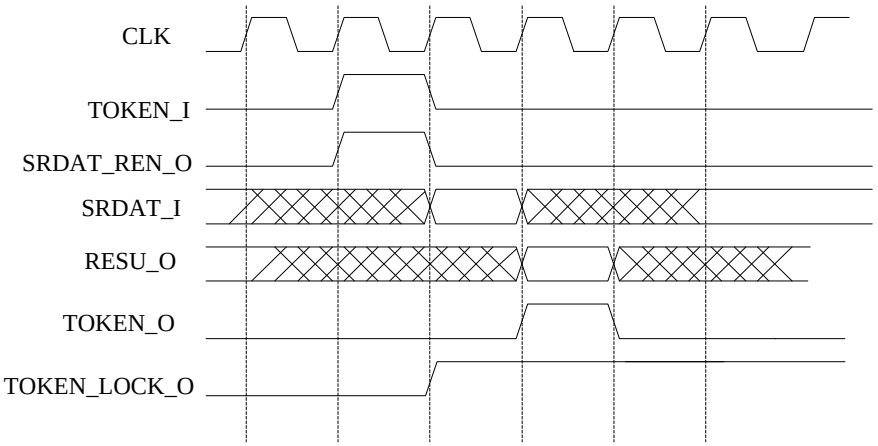


图 7.3　PE 令牌锁存功能

针对不同的算法情况，下面讨论各种情况下令牌在 PE 阵列中流水情况。

无数据相关

此类令牌流比较简单，FIFO 将分组数据和令牌送入第一行 PE，该行 PE 处理完以后，将令牌和数据送入下一行 PE，同时接收下一个分组数据和令牌，如图 7.4 所示，数据和令牌在 PE 阵列中实现流水。

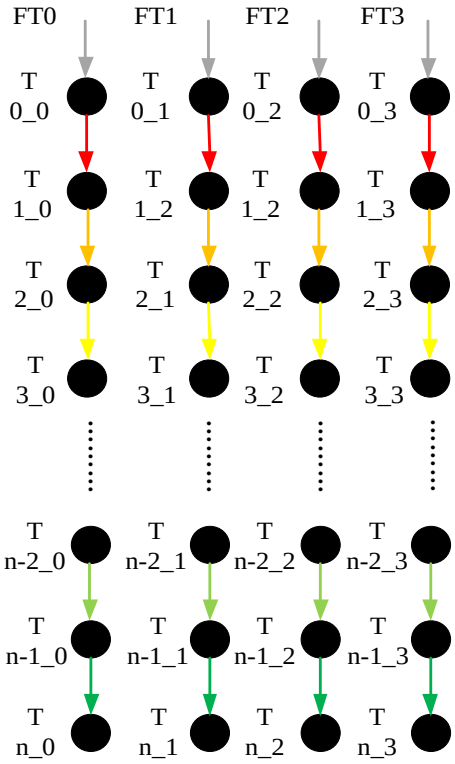
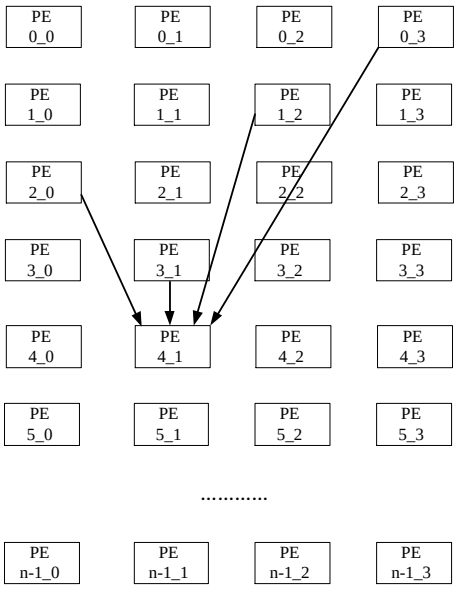


图 7.4　PE 阵列无数据相关令牌流

有数据相关

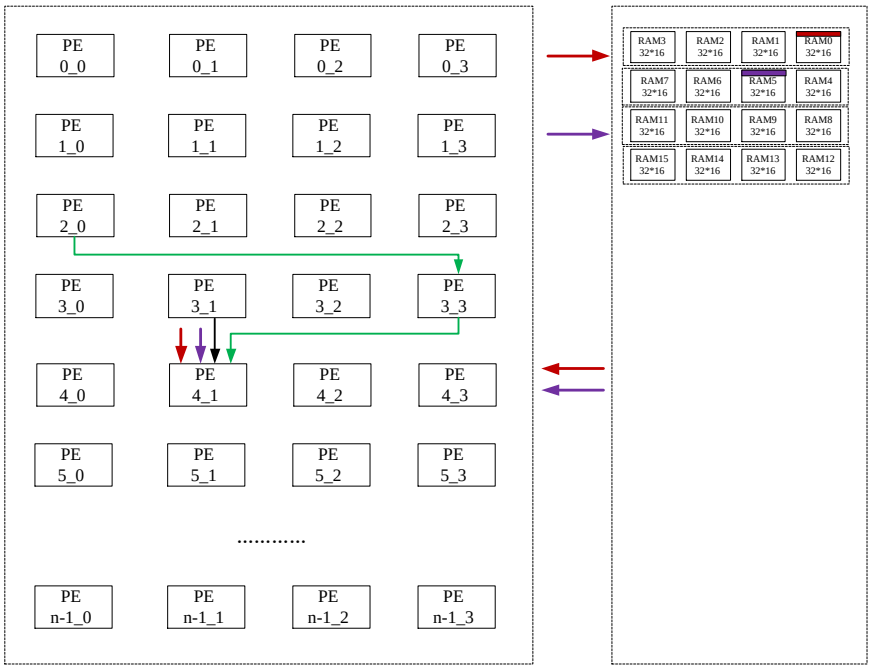
当数据不是顺序从上一行传递到下一行时，尤其当某个 PE 的数据源来自多个 PE 时，此时如果令牌使能流水处理不当的话，就会发生操作数不一致，导致运算出错。

如下图所示，例如，PE（编号为 4_1）的运算需要使用多个 PE 单元输出数据（例如编号为 2_0，3_1，1_2，0_3）。



为解决这样的问题，在算法参数设计时，有 2 种方式解决这样的问题，一是，将需要用的数据透传至使用单元 PE；二是，在行间隔 1 或者 1 以上的单元 PE 处，将需要被使用的的数据存储到通用寄存器堆(但不需要送令牌使能至通用令牌使能通道)，因为当 PE(编号为 4_1)接收到 PE(编号为 3_1)的令牌使能后，通用寄存器堆中的数据已经稳定，可以直接取使用。

实现中可以将这 2 种方式结合使用，例如如下图所示，此例中将行间隔达到 2 或者 2 以上的 PE（本例子中编号为 1_2 和 0_3 的 PE）输出数据送入通用寄存器堆，间隔 1 行的 PE（本例中编号为 2_0）的数据采用透传的方式。



算法完全展开

将算法中的循环在阵列中完全展开，即将整个算法循环复制多份，形成一个整体在阵列上完成映射。

算法完全展开，且无数据相关的令牌流水分两种情况：一，1 行 PE 只实现 1 个分组，如 AES 算法；二， 1 行 PE 可以实现多个分组，如 SM4 算法。（如果 1 行 PE 无法实现 1 个分组的情况放在数据相关类型中进行讨论）。

第一种情况，第一个周期 FIFO 将数据送入第一行 PE 后，同时将 4 个 PE 令牌下发（图中彩色箭头为令牌），如图 7.8 所示。第二个周期，第一行 PE 处理完成后，将数据和令牌送入第二行 PE，同时 FIFO 将新的分组和令牌送入第一行 PE，如图 7.9 所示，以此类推处理。

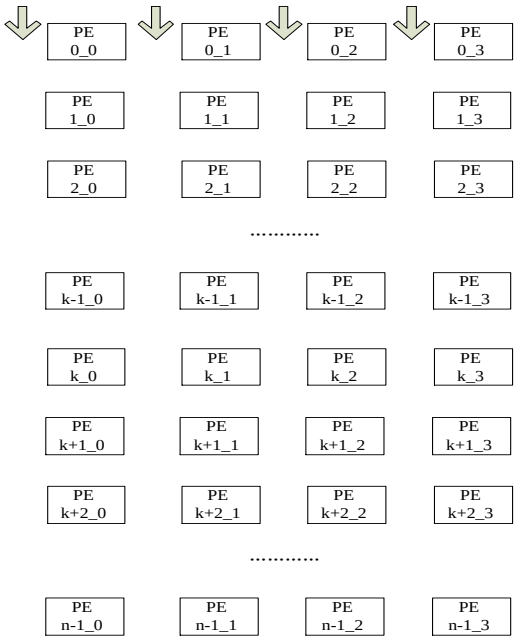


图 7.8　AES 算法令牌流 1

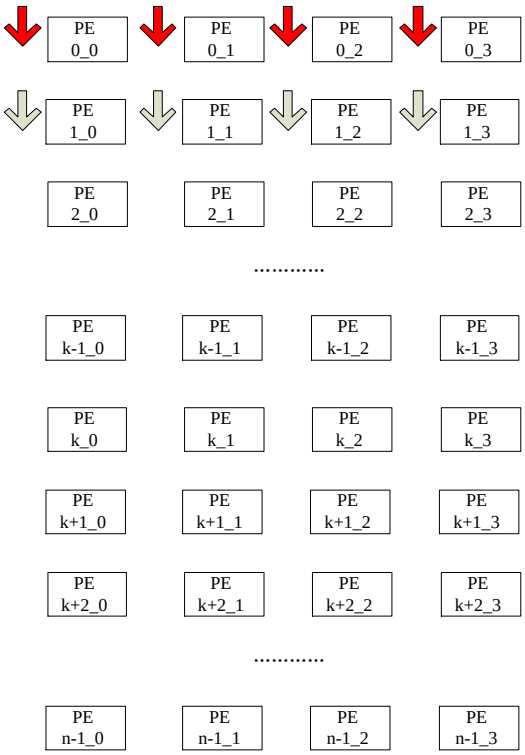


图 7.9　AES 算法令牌流 2

第二种情况，第一个周期 FIFO 将数据送入第一行 PE 的第 0 和 1 号 PE 后，同时将 2 个 PE 令牌下发（图中彩色箭头为令牌），如图 7.10 所示。第二个周期，第一行的第 0 和 1 号 PE 处理完成后，将数据和令牌送入第二行的第 0 和 1 号 PE，同时 FIFO 将新的分组和令牌送入第一行的第 0 和 1 号 PE，如图 7.11 所示。

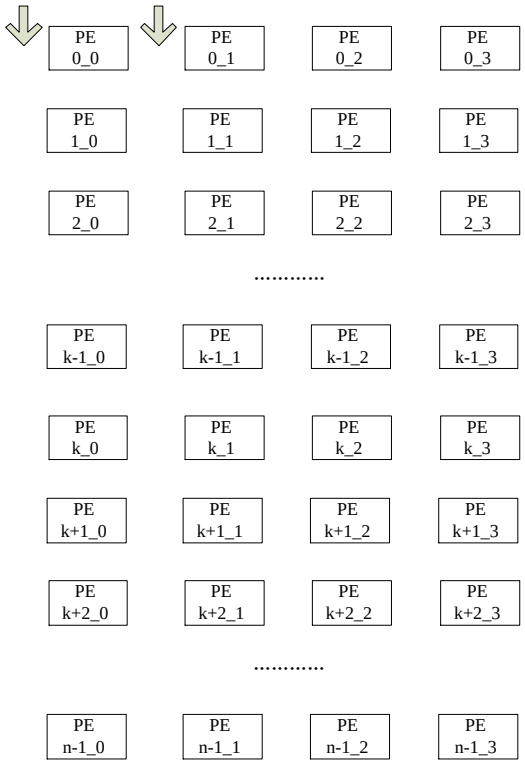


图 7.10　SM4 算法令牌流 1

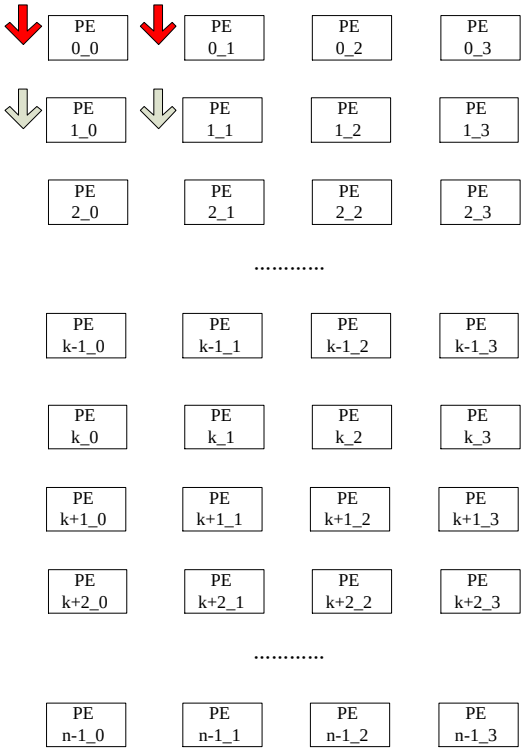


图 7.11　SM4 算法令牌流 2

第 n 个周期后，如图 7.12 所示，第一个分组令牌来到最后一行（第 n 行），PE(编号 n-1_0 和 n-1_1)处理完成后，将数据送入 GPRF，实现和下一令牌接收 PE 的数据交互。

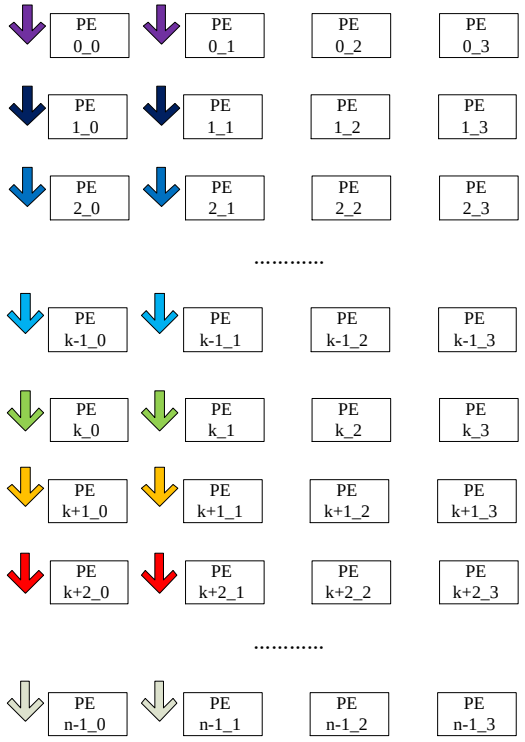


图 7.12　SM4 算法令牌流 3

下一令牌接收 PE（编号为 0_2 和 0_3）接收到令牌后（灰色令牌），如图 7.13 所示，（由于需要从 GPRF 取数，因此该令牌在此 PE 中停留 2 个周期），发出读 GPRF 请求，接收到 GPRF 数据以后，如图 7.14，进行运算，同时接收到了下一个分组的令牌（红色令牌），发出读 GPRF 请求。数据处理完成以后，将数据和令牌（灰色令牌）送出，如图 7.15 所示。

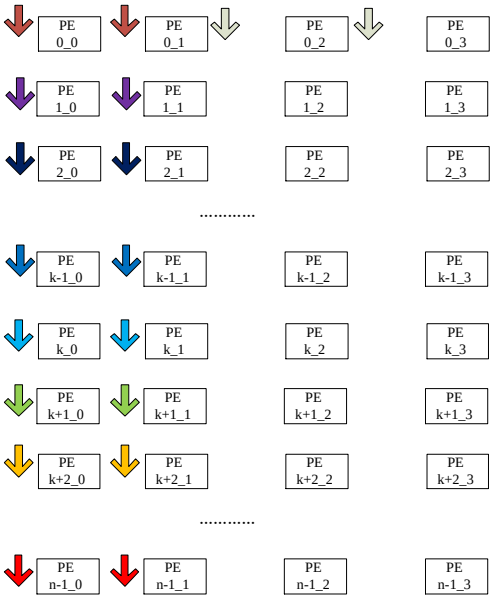


图 7.13　SM4 算法令牌流 4

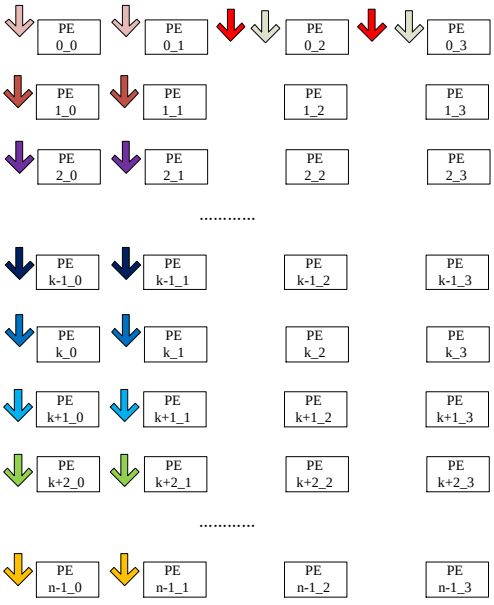


图 7.14　SM4 算法令牌流 5

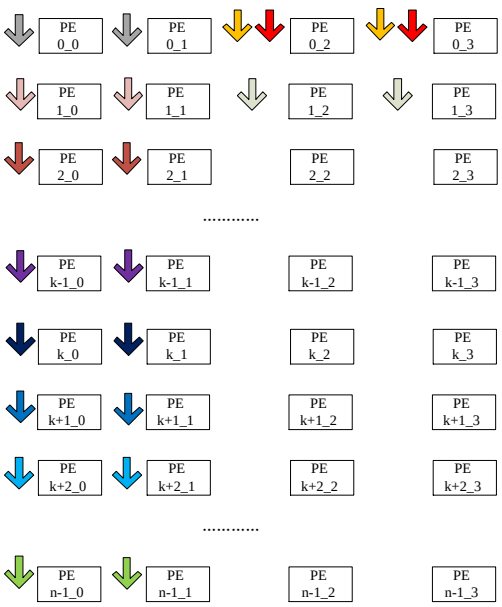
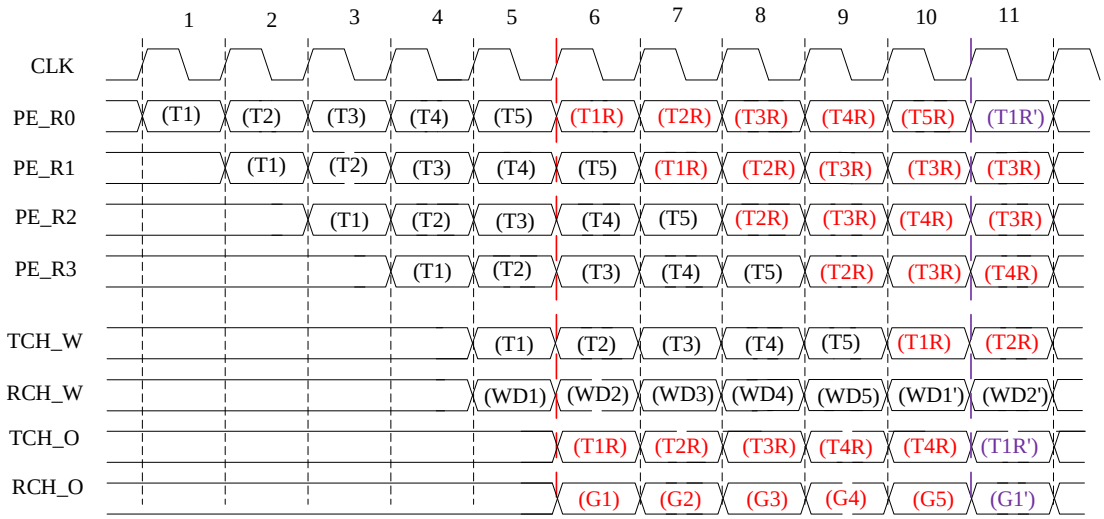


图 7.15　SM4 算法令牌流 6

算法有循环不需要更换参数



算法无循环需要更换参数