

# Vertex Programs and Register Combiners on NV2x

Kim Steen Petersen

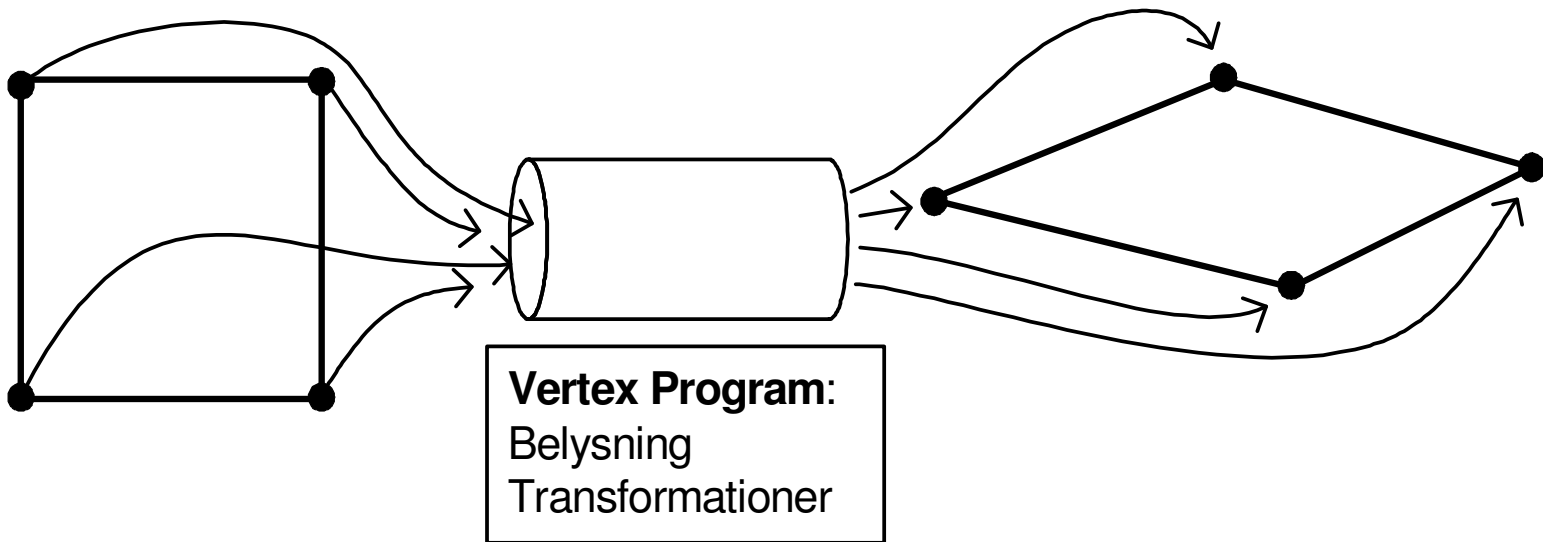
Bjarke Jakobsen

IMM

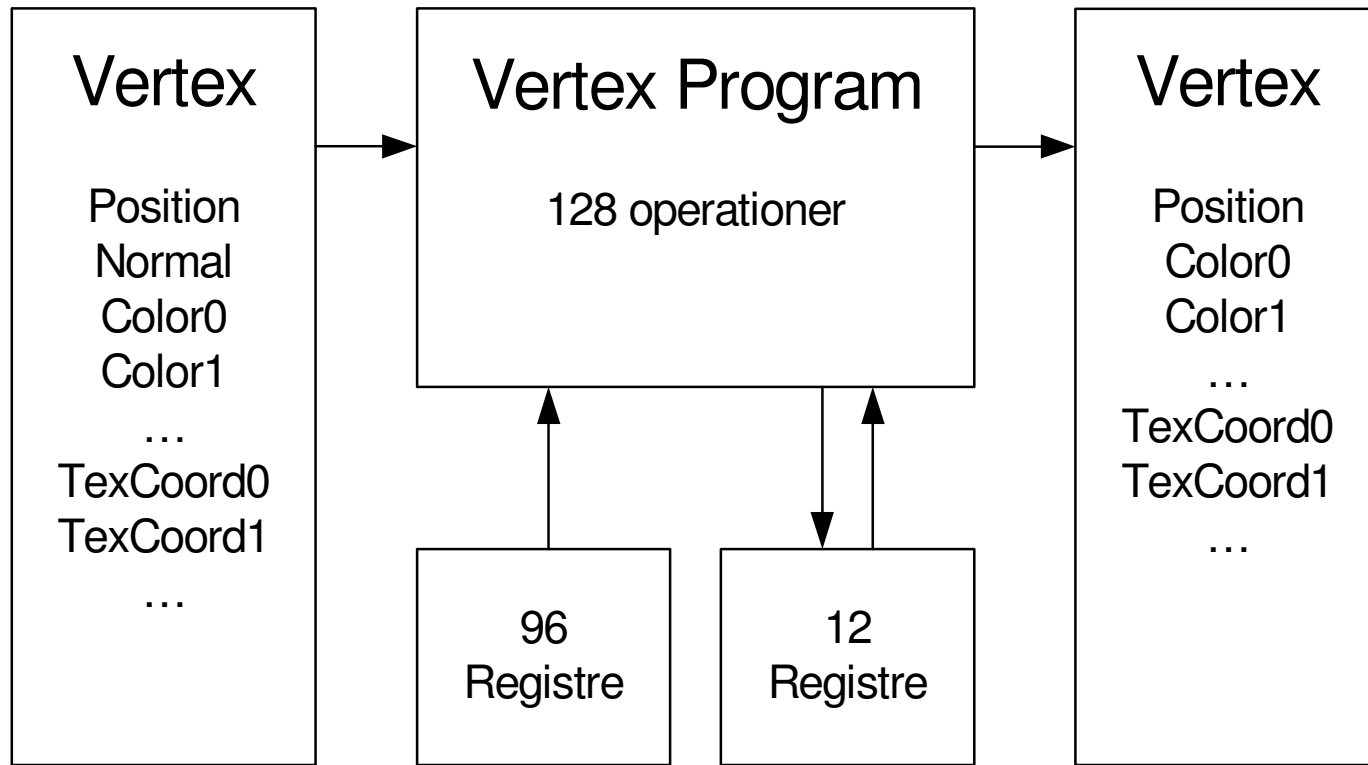
## Programmering af hardware idag

- OpenGL på NV2x
- nvParse
  - Vertex Program
  - Texture Shader
  - Register Combiner

# Vertex Program



# Vertex Program



# Vertex Program

## OpenGL syntax

---

```
glProgramParameter4fNV(...);  
glTrackMatrixNV (...);  
nvparse("!!VP1.0 ...");  
glEnable(GL_VERTEX_PROGRAM_NV);  
glBegin (...);  
glVertexAttrib4fvNV (...);  
glVertex3f (...);  
  
...  
glEnd();  
glDisable(GL_VERTEX_PROGRAM_NV);
```

---

# Vertex Program

## Instruktioner

MOV	$res = arg0$
ADD	$res = arg0 + arg1$
MUL	$res = arg0 * arg1$
MAD	$res = arg0 * arg1 + arg2$
RCP	$res.* = 1/arg0.x$
RSQ	$res.* = 1/\sqrt{arg0.x}$
DP3	$res.* = arg0.xyz \cdot arg1.xyz$
DP4	$res.* = arg0.xyzw \cdot arg1.xyzw$
MIN	$res = MIN(arg0, arg1)$
MAX	$res = MAX(arg0, arg1)$

# Vertex Program

## Transformation of vertex

---

!! *VP1.0*

# Transforming of vertex

# *v[OPOS]* = input vertex position

# *o[HPOS]* = output vertex position

# *c[0] - c[3]* = transformation matrix

**DP4** *o[HPOS].x, c[0], v[OPOS];*

**DP4** *o[HPOS].y, c[1], v[OPOS];*

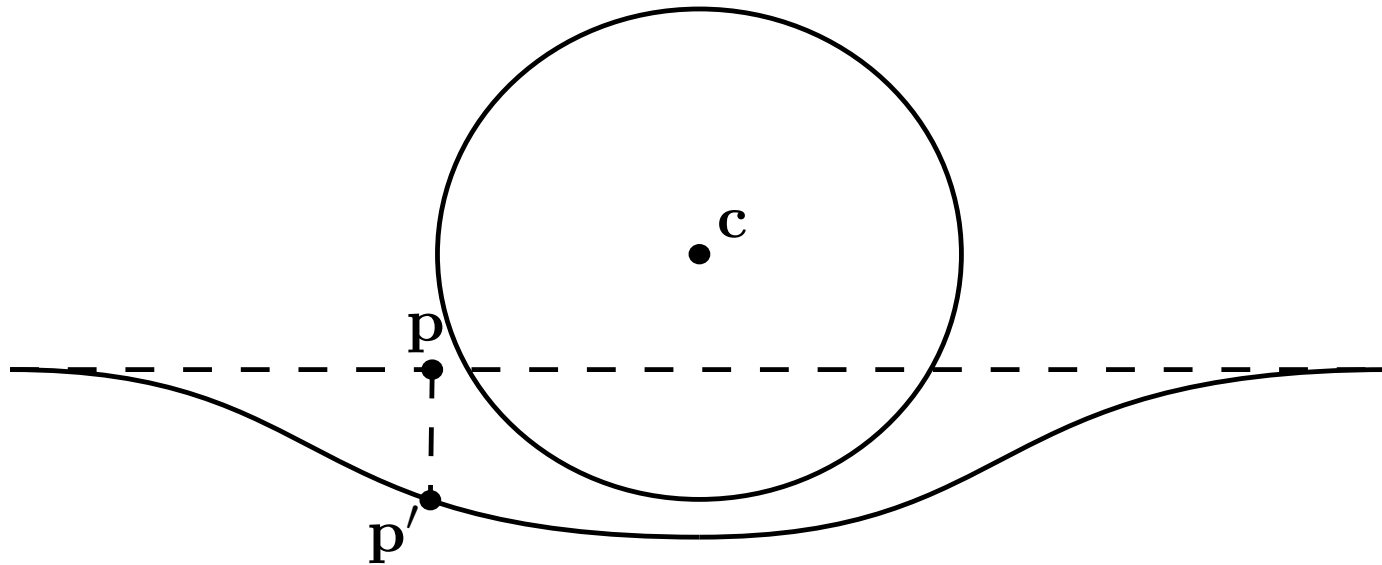
**DP4** *o[HPOS].z, c[2], v[OPOS];*

**DP4** *o[HPOS].w, c[3], v[OPOS];*

**END;**

# Vertex Program

Eksempel

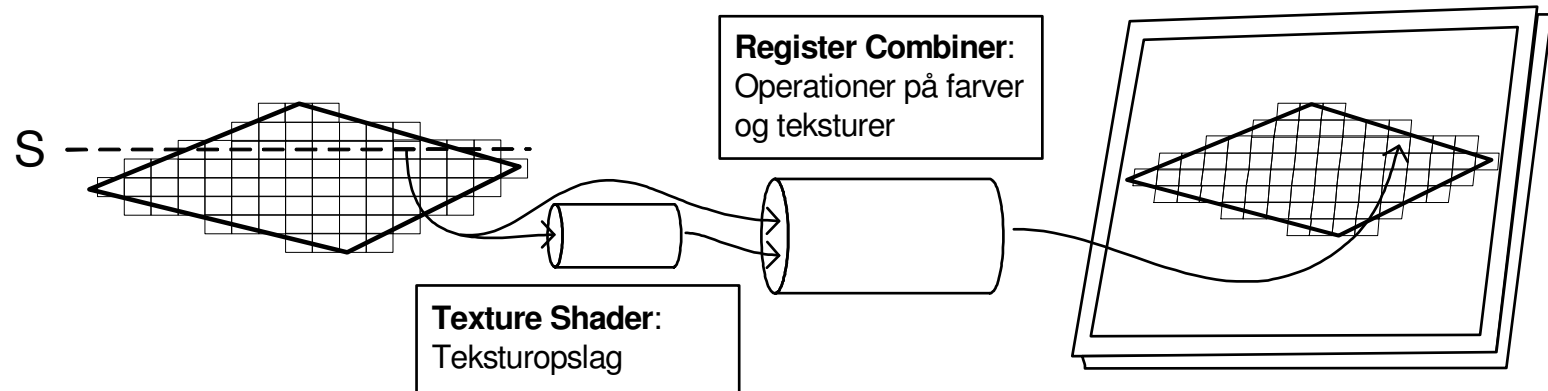


$$p' = f(|cp|)$$

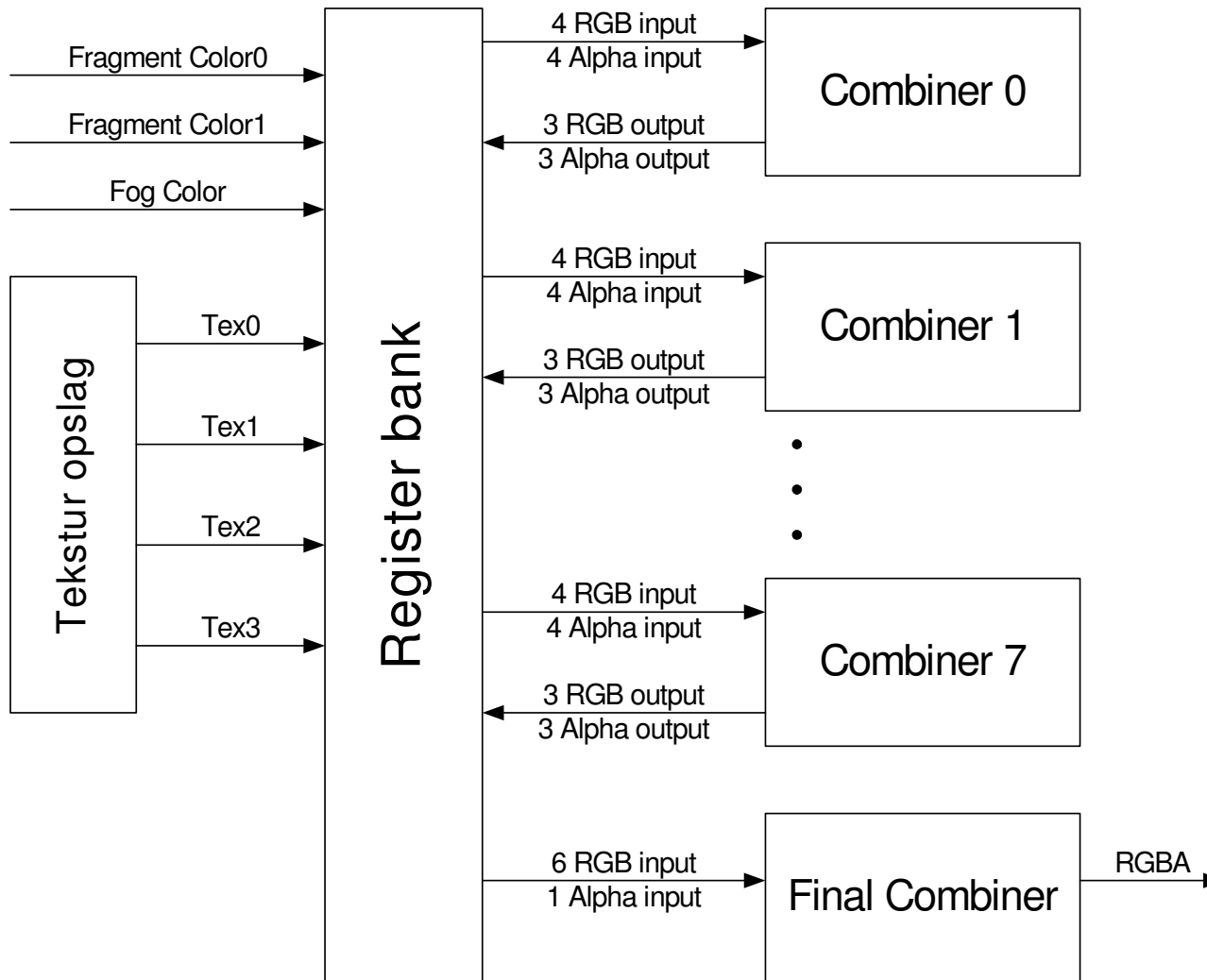
# Vertex Program

```
!! VP1.0
MOV   R0,      v[OPOS];
MOV   R8,      c[95];
ADD   R3.xy,   c[94],    - R0;
DP3   R3,      R3,      R3;
RSQ   R3,      R3.x;
RCP   R3,      R3.x;
MIN   R3,      R3,      c[92].z;
MAX   R3,      R3,      c[93].x;
MUL   R3,      R3,      c[92].w;
ADD   R3,      R3,      -c[93].x;
MUL   R4,      R3,      R3;
MUL   R5,      R4,      R3;
MUL   R4,      R4,      c[93].w;
MAD   R5,      R5,      -c[93].z,    R4;
ADD   R5,      R5,      -c[93].y;
MOV   R2,      c[94].z;
ADD   R2,      c[92].x,    - R2;
MAX   R2,      R2,      c[92].w;
MUL   R0.z,    R5,      R2;
DP4   R1.x,    c[0],     R0;
DP4   R1.y,    c[1],     R0;
DP4   R1.z,    c[2],     R0;
DP4   R1.w,    c[3],     R0;
MOV   o[COL0], v[COL0];
MOV   o[HPOS], R1;
END
```

# Register Combiner



# Register Combiner



# Vertex Program

## OpenGL syntax

---

```
nvparse("!!RC1.0 ...");
```

```
glEnable(GL_REGISTER_COMBINERS_NV);
```

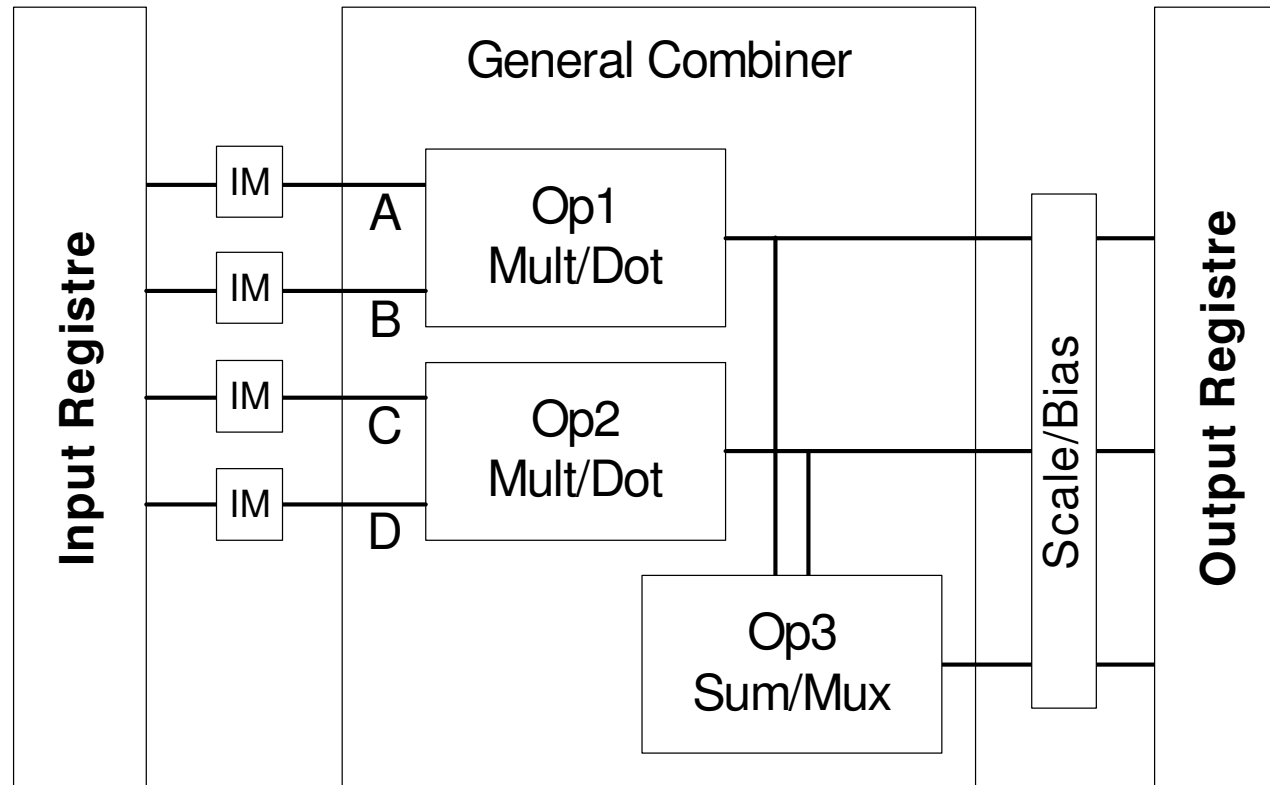
```
drawScene();
```

```
glDisable(GL_REGISTER_COMBINERS_NV);
```

---

# Register Combiner

## General Combiner



# Register Combiner

## Eksempel på General Combiner

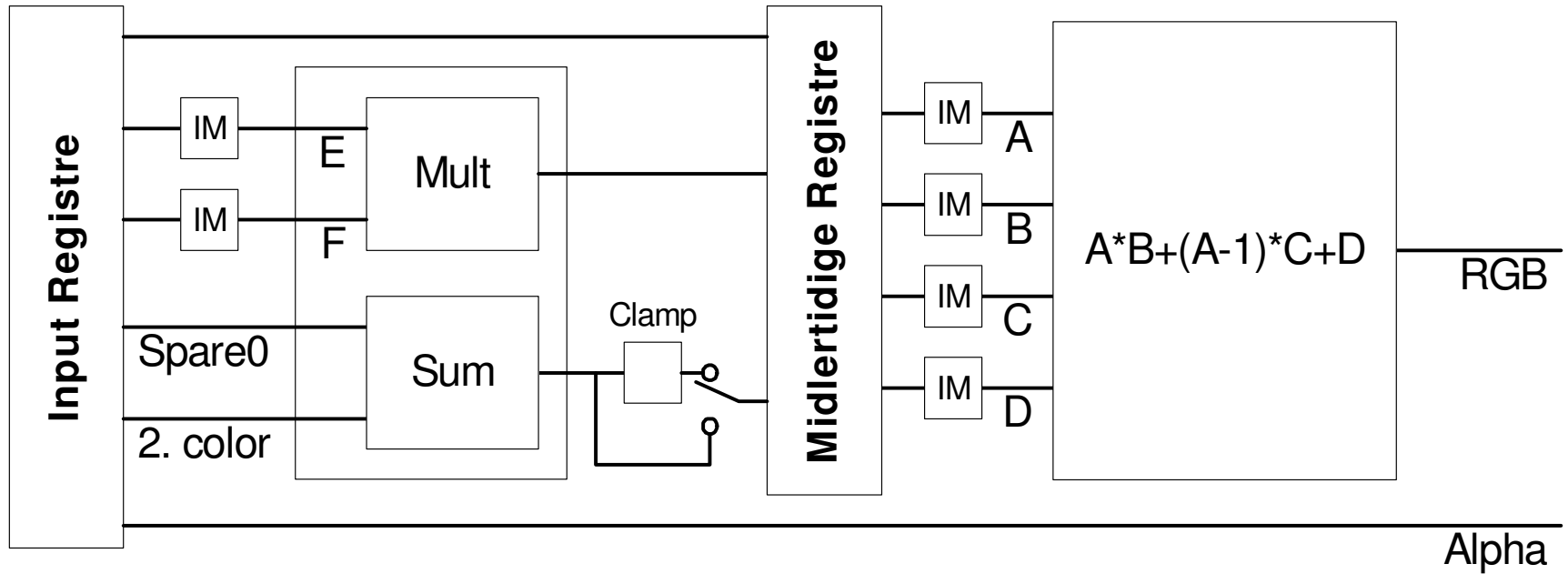
---

```
{  
  rgb  
  {  
    const0 = (.5,.8,1,1);  
    discard = tex1*const0;  
    discard = -col0.a*tex0;  
    spare1 = sum();  
    scale_by_two();  
  }  
}
```

---

# Register Combiner

## Final Combiner



# Register Combiner



## Final Combiner

---

```
final_product = tex1*unsigned_invert(col0);  
out.rgb = lerp(spare1,col0,tex3.a)+final_product;  
out.a = unsigned_invert(zero);
```

---

# Register Combiner

Per Pixel Division vha. Taylorudvikling

$$\begin{aligned}\frac{1}{1-x} &= 2 + 4\left(x - \frac{1}{2}\right) + 8\left(x - \frac{1}{2}\right)^2 + 16\left(x - \frac{1}{2}\right)^3 + \dots \\ &= 2 \sum_{i=0}^{\infty} (2x - 1)^i \quad , 0 < x < 1\end{aligned}$$

# Register Combiner

Per Pixel Division vha. Taylorudvikling

$$\frac{1}{1-x} = 2 + 4\left(x - \frac{1}{2}\right) + 8\left(x - \frac{1}{2}\right)^2 + 16\left(x - \frac{1}{2}\right)^3 + \dots$$

$$= 2 \sum_{i=0}^{\infty} (2x - 1)^i, \quad 0 < x < 1$$

$$\frac{A}{B} = 2A \sum_{i=0}^{\infty} (2(1-B) - 1)^i = 2A \prod_{i=0}^{\infty} (1 + (1-2B)^{2^i})$$

# Register Combiner

## Rekursionsformel

$$R_0 = 2A$$

$$Q_0 = 1 - 2B$$

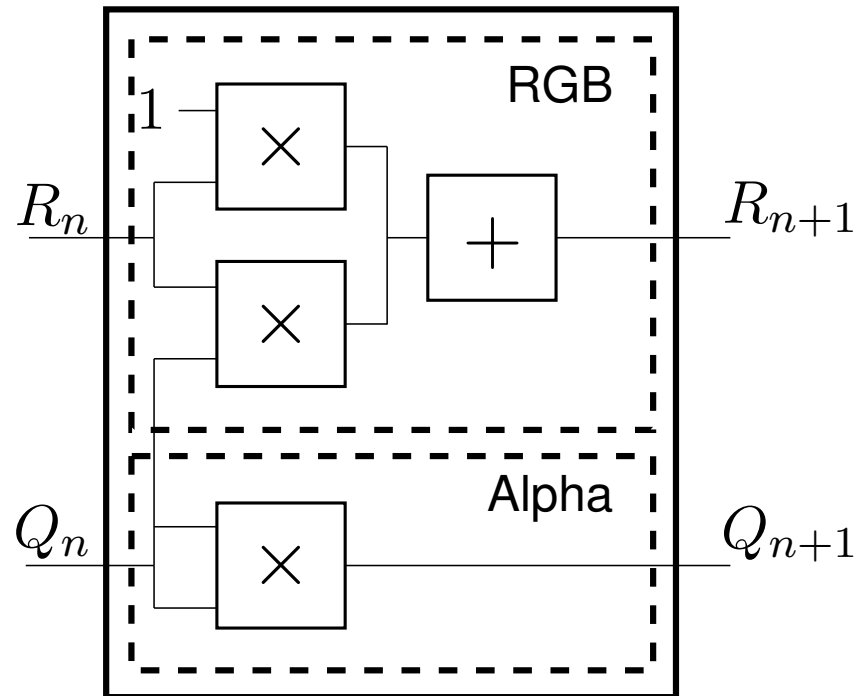
$$R_{n+1} = R_n + R_n \cdot Q_n$$

$$Q_{n+1} = Q_n \cdot Q_n$$

$$\lim_{n \rightarrow \infty} R_n = \frac{A}{B}$$

# Register Combiner

”Pseudokode” for Combiner



# Register Combiner

## Kode for Divisions Combiner

---

```
rgb {  
    discard = spare1;  
    discard = spare1*spare1.a;  
    spare1 = sum();  
}  
alpha {  
    spare1 = spare1*spare1;  
}
```

---

# Spørgsmål