StarCore : Computing Correlation Function

Young Won Lim
5/28/16

Please send corrections (or suggestions) to i.

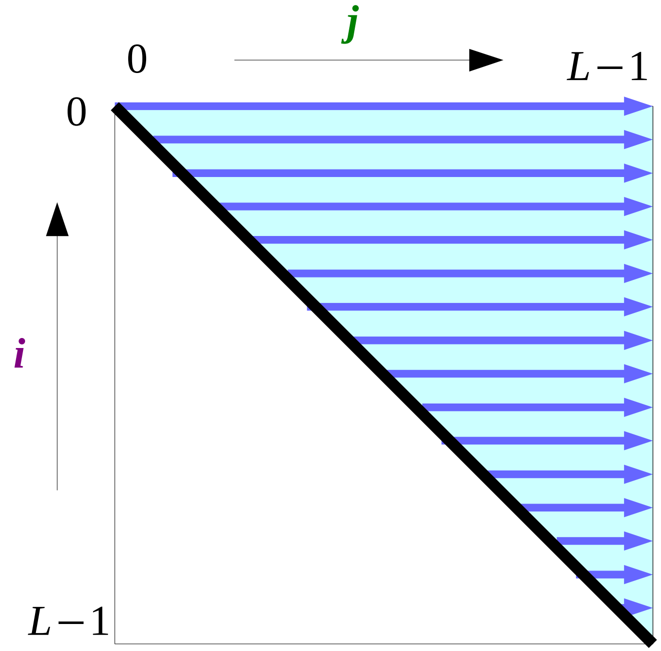
This document was produced by using OpenOffice.

Based on

Cross Correlation
http://cache.freescale.com/files/dsp/doc/app_note/AN2266.pdf

Correlation Code

```
L_max = 0;
for (i = L-1; i >= 0; i--) {
        Acc = 0;
        for (j = i; j < L; j++)
                Acc = L_mac(Acc, x[j], h[j-i]);
        y[i] = Acc;
        Acc = L_abs(Acc);
        if ( Acc > L_max ) {
                L_max = Acc;
        }
}
```



$$y[i]=\sum_{j=i}^{L-1} x[j]h[j-i]$$

L_max
L_mac
L_abs

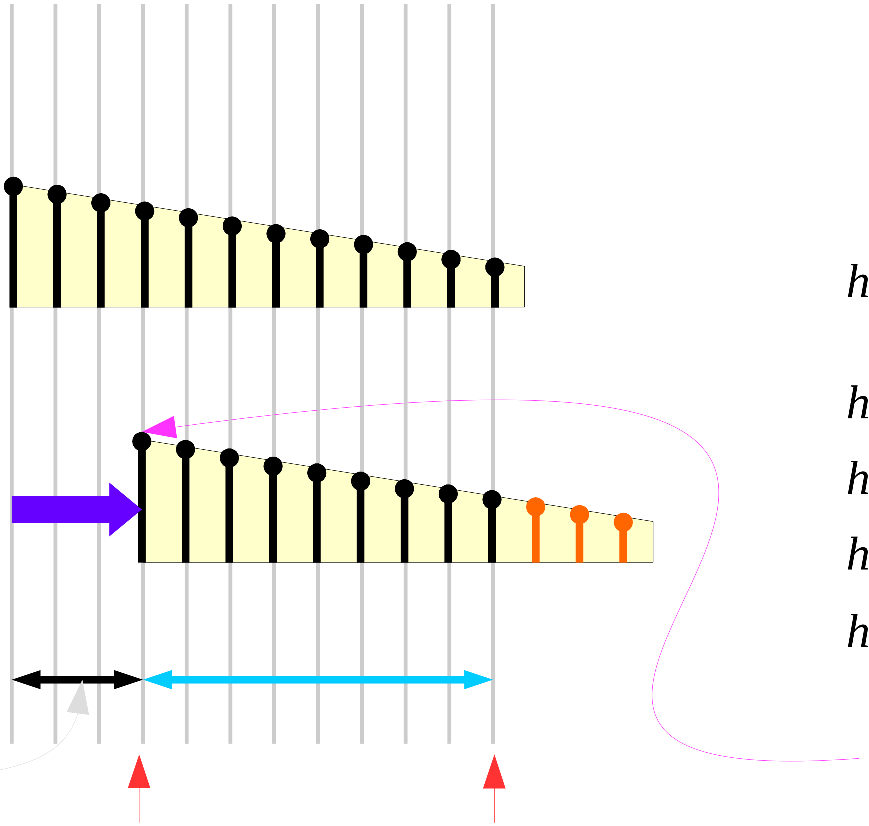
h[j] and h[0]

$$y[i]=\sum_{j=i}^{L-1} x[j]h[j-1]$$

$h[j]$

$h[j-3]$

$i = 3$



$j=i$ $j=L-1$

$h[j-i] = 0$
$(j < i)$

$h[j]$

$h[j-3]$ $h[j]$

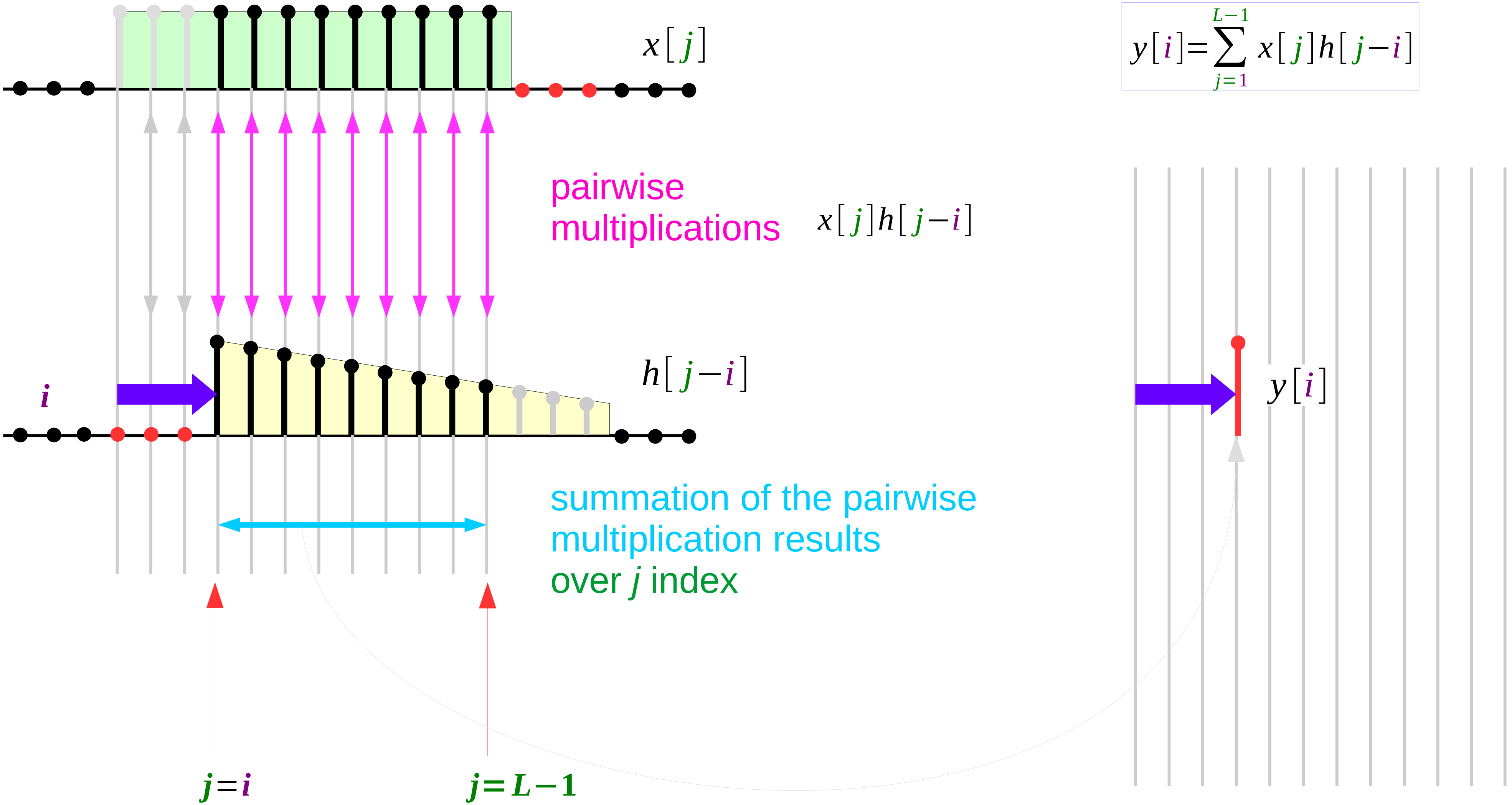
$h[j-i]$ shifted version of $h[j]$

$h[n-a]$ $h[n]$

$h(t-a)$ $h(t)$

$h[j-i] = h[0]$ *when* $j = i$

Correlation y[*i*] – for a given *i*

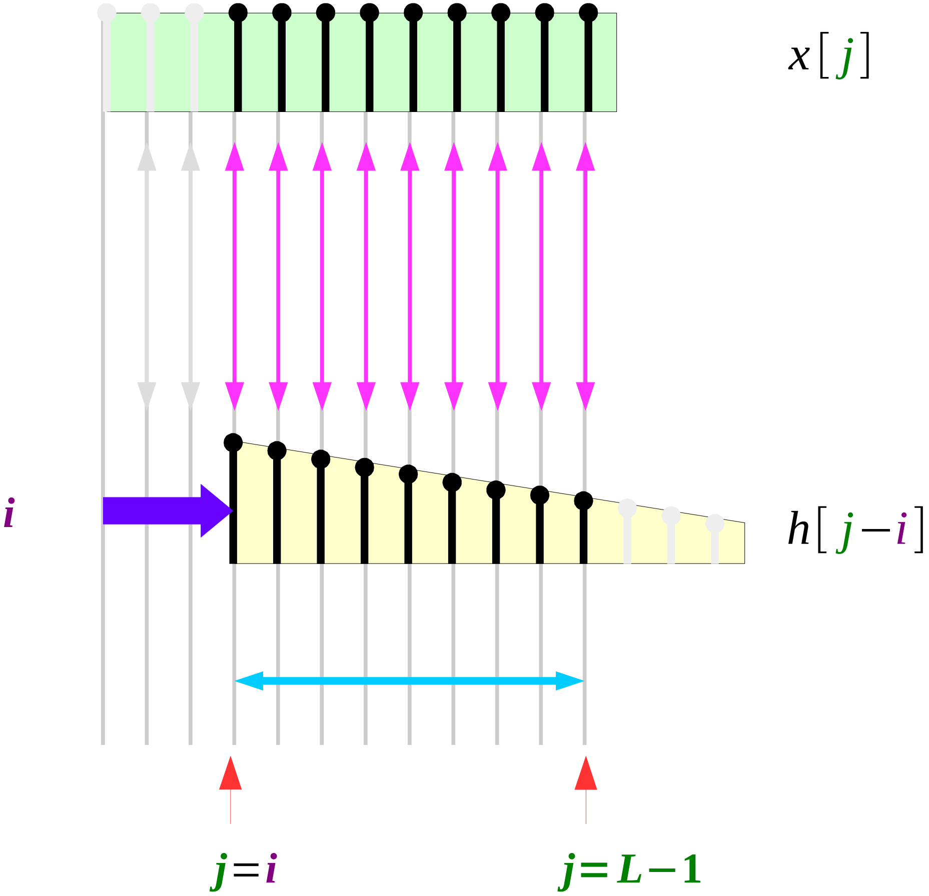


$$y[i]=\sum_{j=1}^{L-1} x[j]h[j-i]$$

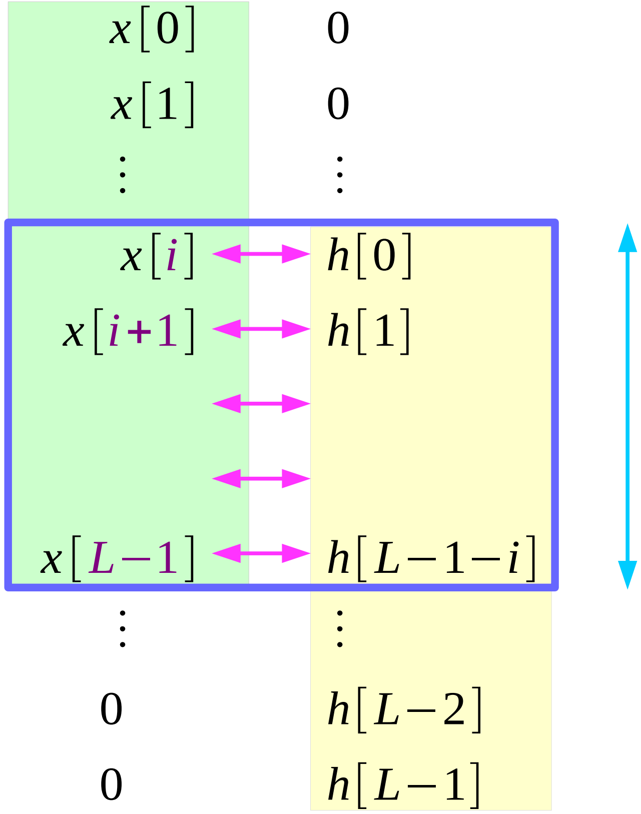
pairwise multiplications $x[j]h[j-i]$

summation of the pairwise multiplication results over *j* index

Summing pairwise multiplications



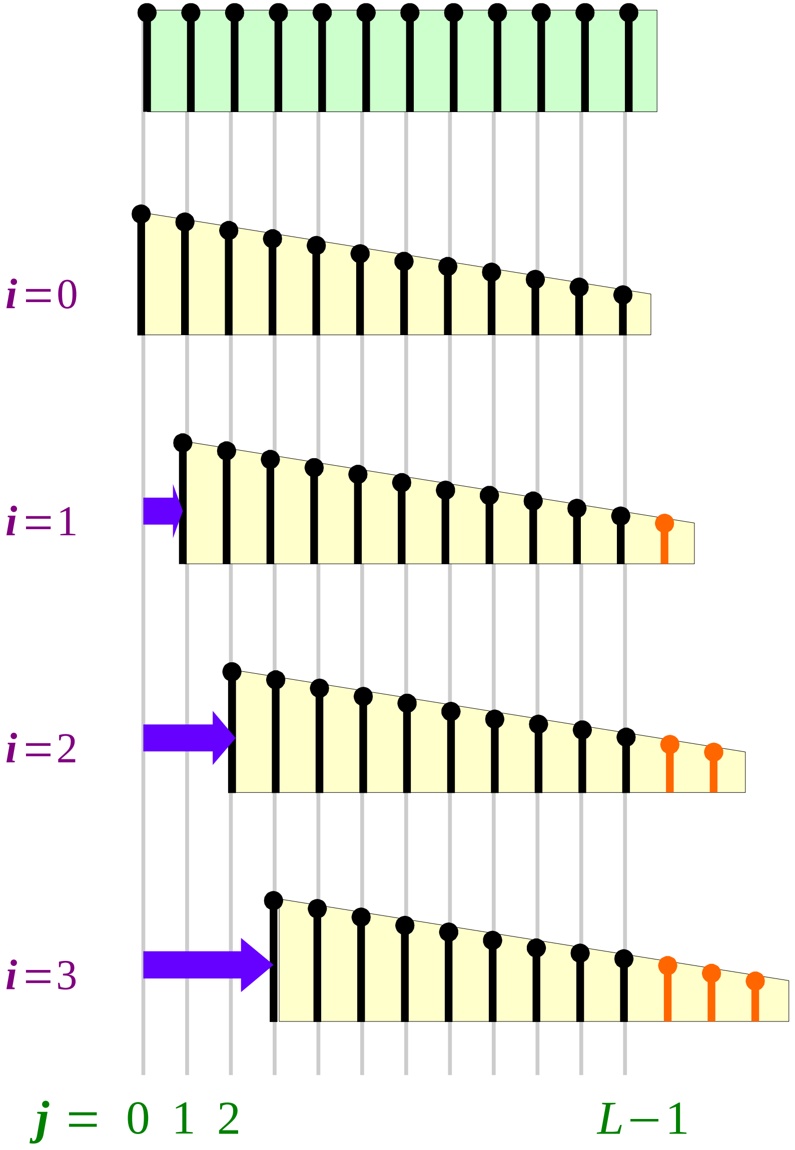
$$y[1]=\sum_{j=1}^{L-1} x[j]h[j-1]$$



$x[0]$	0
$x[1]$	0
$\vdots$	$\vdots$
$x[i]$	$h[0]$
$x[i+1]$	$h[1]$
$x[L-1]$	$h[L-1-i]$
$\vdots$	$\vdots$
0	$h[L-2]$
0	$h[L-1]$

Cross Correlation Function y[*i*]

$$y[i] = \sum_{j=i}^{L-1} x[j]\,h[j-i]$$



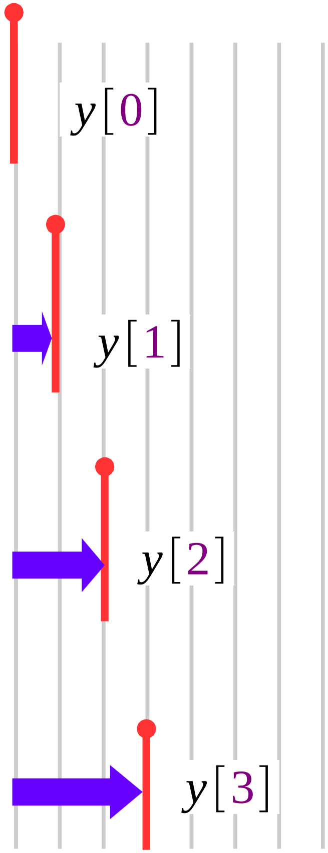
$x[j]$

$h[j-0]$ $$\sum_{j=0}^{L-1} x[j]\,h[j-i]$$

$h[j-1]$ $$\sum_{j=1}^{L-1} x[j]\,h[j-i]$$

$h[j-2]$ $$\sum_{j=2}^{L-1} x[j]\,h[j-i]$$

$h[j-3]$ $$\sum_{j=3}^{L-1} x[j]\,h[j-i]$$



Loop Unroll to OL with IL0, IL1, IL2, IL3

```
Lx = 0;
for (i = L-1; i >= 0; i--) {
        Acc = comp_y(i);    ➡
        Acc = L_abs(Acc);
        if ( Acc > Lx ) {
                Lx = Acc;
        }
}
```

```
Acc = 0;
for (j = i; j < L; j++)
        Acc = L_mac(Acc, x[j], h[j-i]);
y[i] = Acc;
```

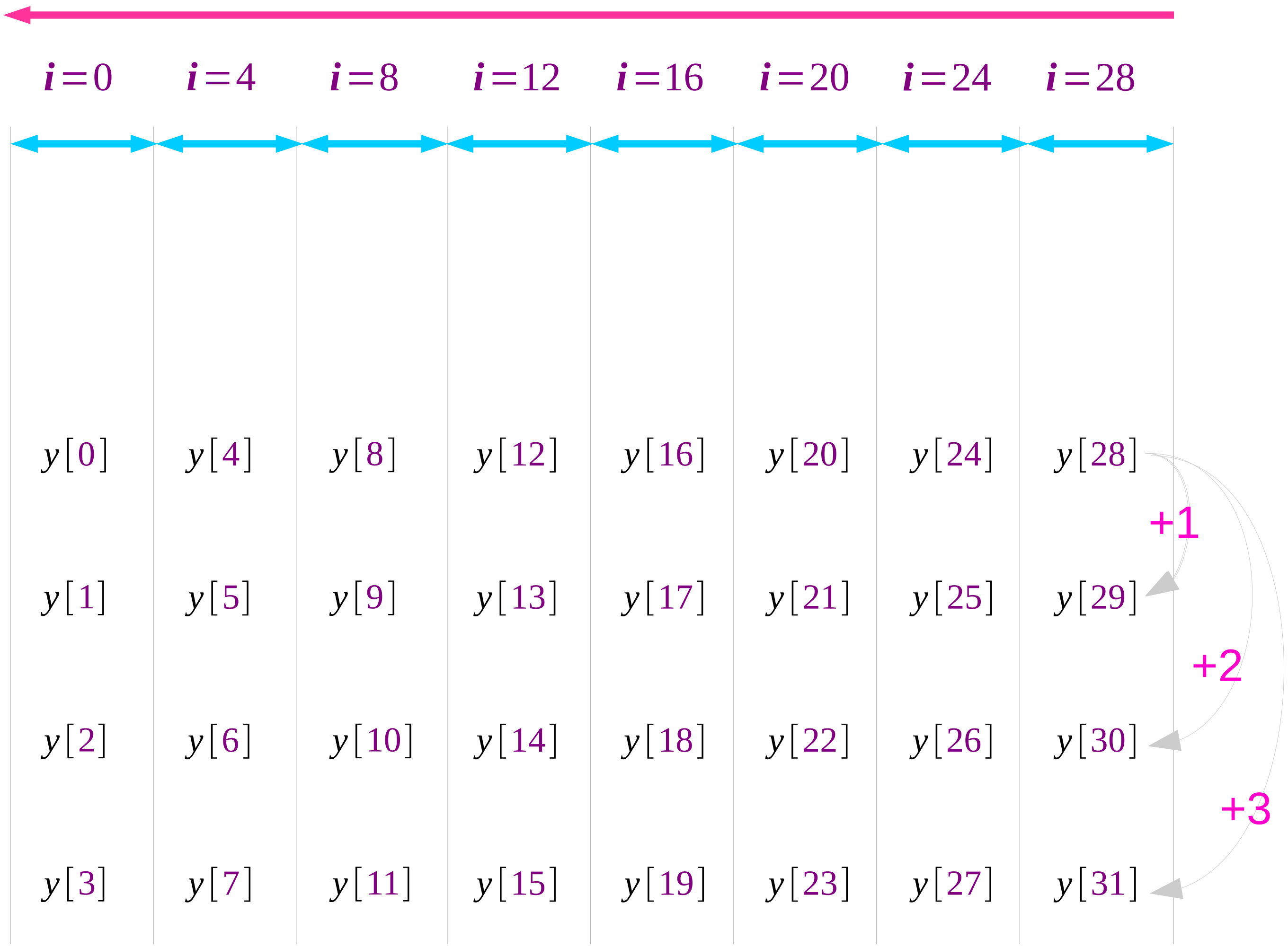
```
Lx0 = Lx1 = Lx2 = Lx3= 0;
for (i = L-4; i >= 0; i-=4) {
  IL0 ➡  Lx0 = find_max_y(i, 0);  ➡
  IL1 ➡  Lx1 = find_max_y(i, 1);  ➡
  IL2 ➡  Lx2 = find_max_y(i, 2);  ➡
  IL3 ➡  Lx3 = find_max_y(i, 3);  ➡
}

Lx = calc_max(Lx0,Lx1,Lx2,Lx3););
```

```
Lxi = 0;

  Acck = comp_y(i+k);
  Acck = L_abs(Acc);
  if ( Acck > Lxk ) {
          Lxk = Acc;
  }
```

New Inner Loops : IL0, IL1, IL2, IL3

```
for (i = L-4; i >= 0; i-=4) {
```

$$y[i]=\sum_{j=i}^{L-1} x[j]h[j-i]$$



	$i=0$	$i=4$	$i=8$	$i=12$	$i=16$	$i=20$	$i=24$	$i=28$
IL0	$y[0]$	$y[4]$	$y[8]$	$y[12]$	$y[16]$	$y[20]$	$y[24]$	$y[28]$
IL1	$y[1]$	$y[5]$	$y[9]$	$y[13]$	$y[17]$	$y[21]$	$y[25]$	$y[29]$
IL2	$y[2]$	$y[6]$	$y[10]$	$y[14]$	$y[18]$	$y[22]$	$y[26]$	$y[30]$
IL3	$y[3]$	$y[7]$	$y[11]$	$y[15]$	$y[19]$	$y[23]$	$y[27]$	$y[31]$

```
}
```

Partitioning y[*i*], $i = 0, \dots, L-1$

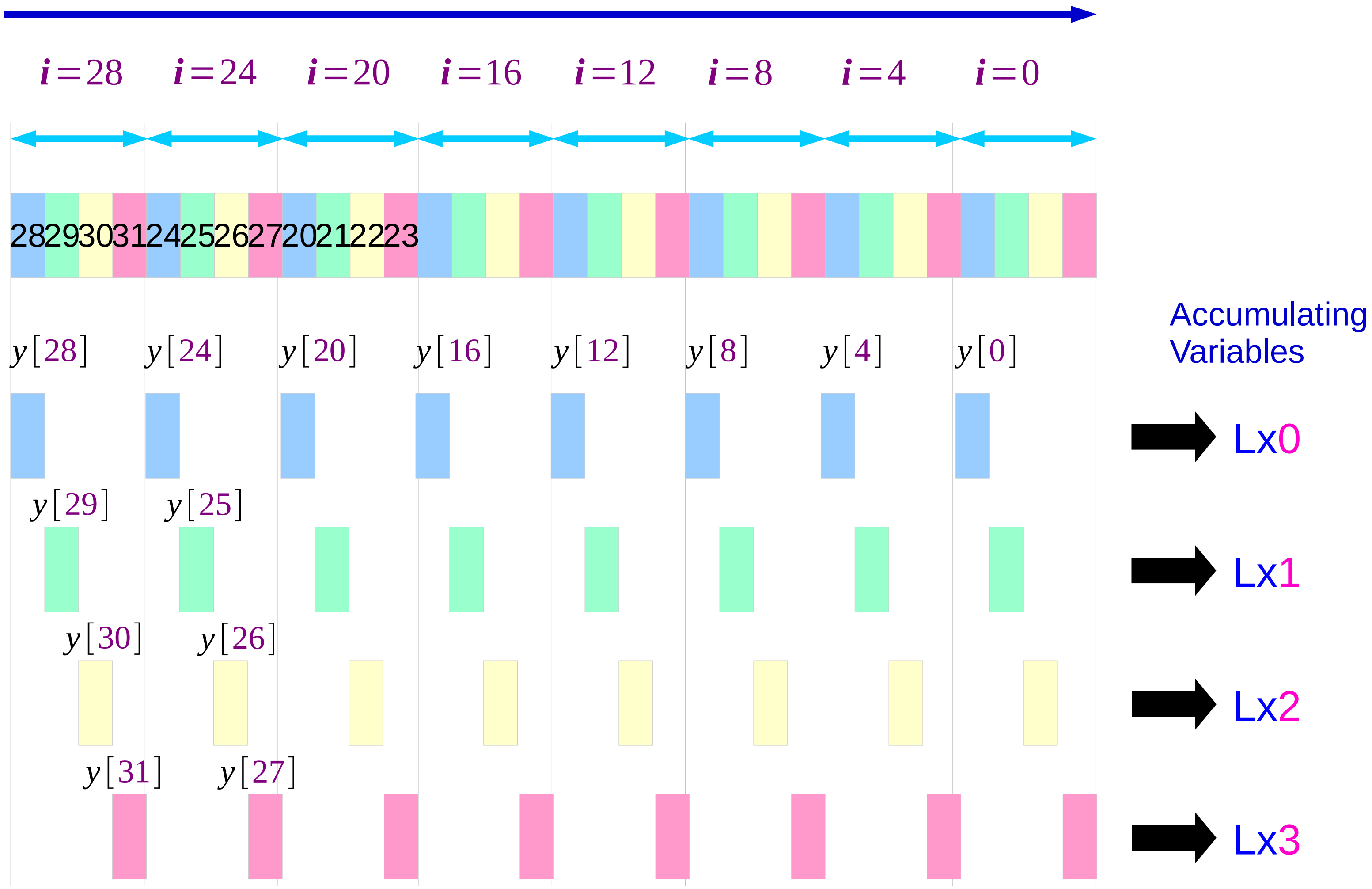
$$y[i] = \sum_{j=i}^{L-1} x[j]h[j-i]$$

$k=0$ sample

$k=1$ sample

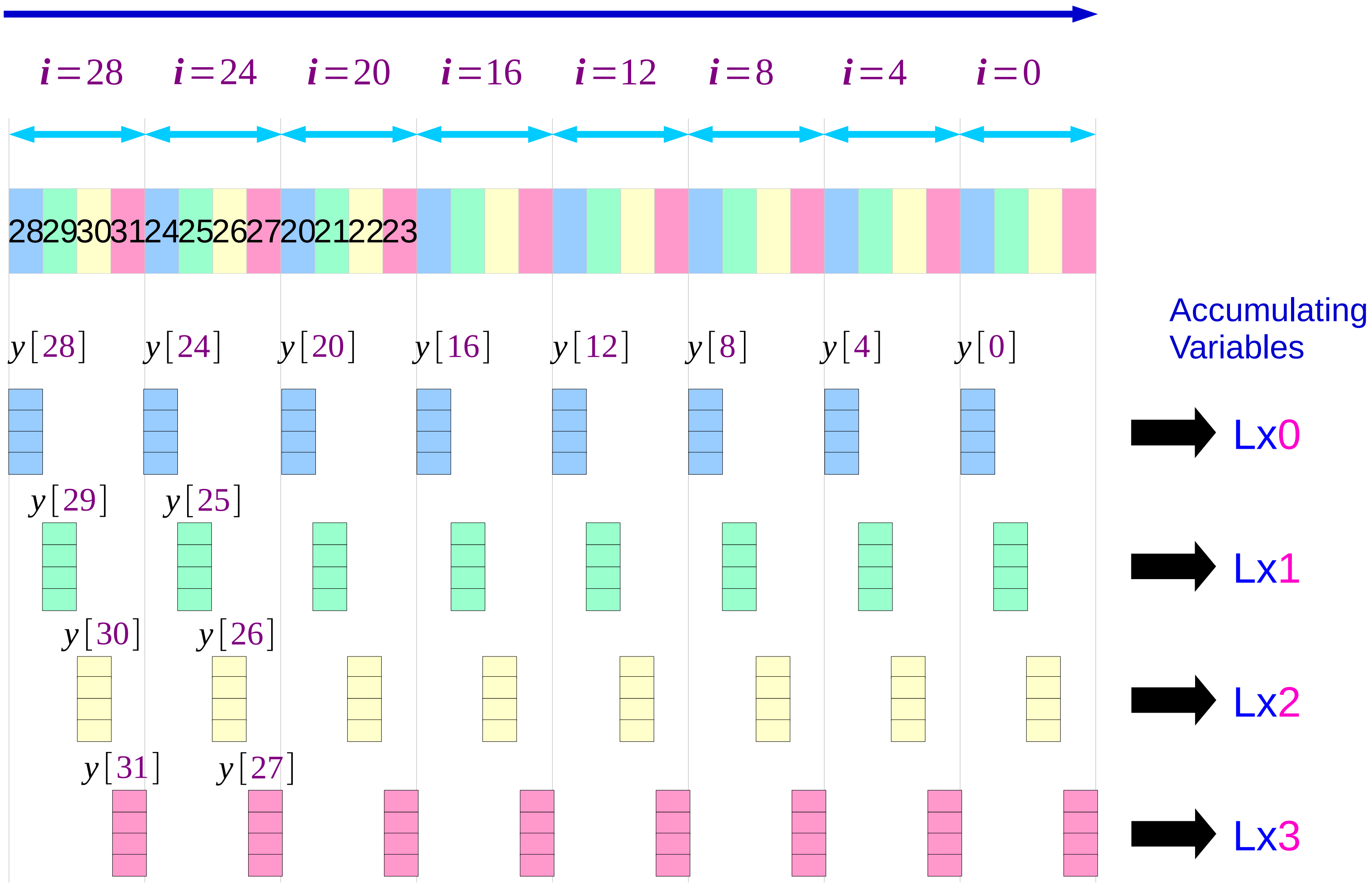
$k=2$ sample

$k=3$ sample



Partitioning each y[*i*]

$$y[i] = \sum_{j=i}^{L-1} x[j]h[j-i]$$



$i=28$ $i=24$ $i=20$ $i=16$ $i=12$ $i=8$ $i=4$ $i=0$
28293031242526272021222 3
$y[28]$ $y[24]$ $y[20]$ $y[16]$ $y[12]$ $y[8]$ $y[4]$ $y[0]$
$y[29]$ $y[25]$
$y[30]$ $y[26]$
$y[31]$ $y[27]$
Accumulating Variables
Lx0
Lx1
Lx2
Lx3

$k=0$ sample

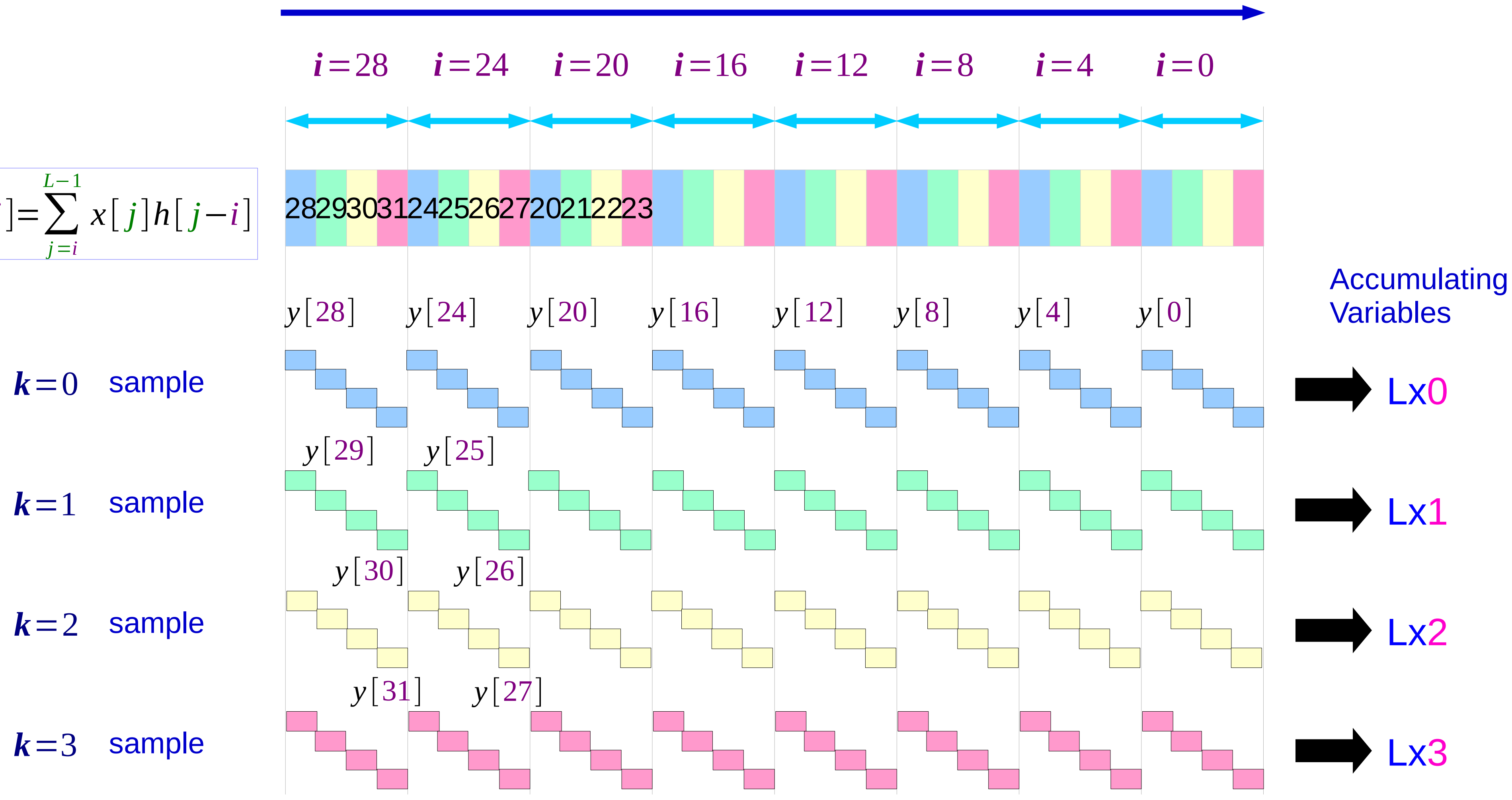
$k=1$ sample

$k=2$ sample

$k=3$ sample

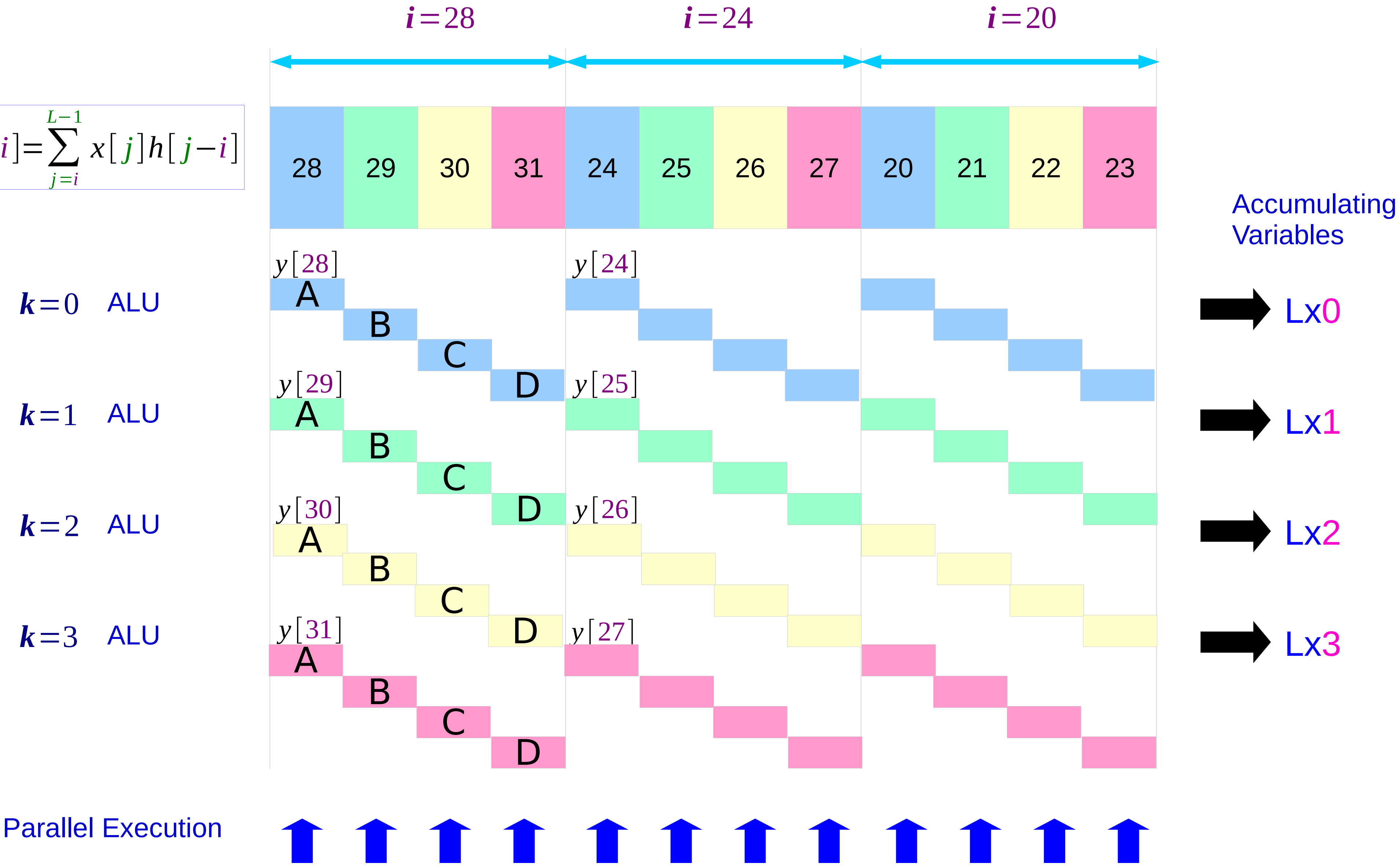
Rearranging partitions

$$y[i] = \sum_{j=i}^{L-1} x[j]\,h[j-i]$$



Detailed View

$$y[i] = \sum_{j=i}^{L-1} x[j]h[j-i]$$



Detailed Code View

```
for (i = L-4; i >= 0; i-=4) {
```

IL0 ➡ $y[i+0] = \sum_{j=i}^{L-1} X[j]h[j-i-0]$

- $\sum \; j = i+0,\; i+4,\; i+8,\; \cdots,\; L-4$ ➡ IL0-A
- $\sum \; j = i+1,\; i+5,\; i+9,\; \cdots,\; L-3$ ➡ IL0-B
- $\sum \; j = i+2,\; i+6,\; i+10,\; \cdots,\; L-2$ ➡ IL0-C
- $\sum \; j = i+3,\; i+7,\; i+11,\; \cdots,\; L-1$ ➡ IL0-D

IL1 ➡ $y[i+1] = \sum_{j=i}^{L-1} X[j]h[j-i-1]$

- $\sum \; j = i+1,\; i+5,\; i+9,\; \cdots,\; L-3$ ➡ IL1-A
- $\sum \; j = i+2,\; i+6,\; i+10,\; \cdots,\; L-2$ ➡ IL1-B
- $\sum \; j = i+3,\; i+7,\; i+11,\; \cdots,\; L-1$ ➡ IL1-C
- $\sum \; j = i+4,\; i+8,\; i+12,\; \cdots,\; L-4$ ➡ IL1-D

IL2 ➡ $y[i+2] = \sum_{j=i}^{L-1} X[j]h[j-i-2]$

- $\sum \; j = i+2,\; i+6,\; i+10,\; \cdots,\; L-2$ ➡ IL1-A
- $\sum \; j = i+3,\; i+7,\; i+11,\; \cdots,\; L-1$ ➡ IL1-B
- $\sum \; j = i+4,\; i+8,\; i+12,\; \cdots,\; L-4$ ➡ IL1-C
- $\sum \; j = i+5,\; i+9,\; i+13,\; \cdots,\; L-3$ ➡ IL1-D

IL3 ➡ $y[i+3] = \sum_{j=i}^{L-1} X[j]h[j-i-3]$

- $\sum \; j = i+3,\; i+7,\; i+11,\; \cdots,\; L-1$ ➡ IL1-A
- $\sum \; j = i+4,\; i+8,\; i+12,\; \cdots,\; L-4$ ➡ IL1-B
- $\sum \; j = i+5,\; i+9,\; i+13,\; \cdots,\; L-3$ ➡ IL1-C
- $\sum \; j = i+6,\; i+10,\; i+14,\; \cdots,\; L-2$ ➡ IL1-D

```
}
```

Accessing h[j] array

```
for (i = L-4; i >= 0; i-=4) {
```

$$y[i+0] = \sum_{j=i}^{L-1} X[j]h[j-i-0] \begin{cases} \sum \; j = i+0,\ i+4,\ i+8,\ \cdots,\ L-4 & \Rightarrow h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4] \\ \sum \; j = i+1,\ i+5,\ i+9,\ \cdots,\ L-3 & \Rightarrow h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3] \\ \sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2 & \Rightarrow h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2] \\ \sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1 & \Rightarrow h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1] \end{cases}$$

$$y[i+1] = \sum_{j=i}^{L-1} X[j]h[j-i-1] \begin{cases} \sum \; j = i+1,\ i+5,\ i+9,\ \cdots,\ L-3 & \Rightarrow h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4] \\ \sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2 & \Rightarrow h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3] \\ \sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1 & \Rightarrow h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2] \\ \sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4 & \Rightarrow h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1] \end{cases}$$

$$y[i+2] = \sum_{j=i}^{L-1} X[j]h[j-i-2] \begin{cases} \sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2 & \Rightarrow h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4] \\ \sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1 & \Rightarrow h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3] \\ \sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4 & \Rightarrow h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2] \\ \sum \; j = i+5,\ i+9,\ i+13,\ \cdots,\ L-3 & \Rightarrow h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1] \end{cases}$$

$$y[i+3] = \sum_{j=i}^{L-1} X[j]h[j-i-3] \begin{cases} \sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1 & \Rightarrow h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4] \\ \sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4 & \Rightarrow h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3] \\ \sum \; j = i+5,\ i+9,\ i+13,\ \cdots,\ L-3 & \Rightarrow h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2] \\ \sum \; j = i+6,\ i+10,\ i+14,\ \cdots,\ L-2 & \Rightarrow h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1] \end{cases}$$

```
}
```

Rearranging for Parallel Execution

```
for (i = L-4; i >= 0; i-=4) {
```

$\sum \; j = i+0,\ i+4,\ i+8,\ \cdots,\ L-4$ → IL0-A

$\sum \; j = i+1,\ i+5,\ i+9,\ \cdots,\ L-3$ → IL1-A

$\sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2$ → IL1-A

$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$ → IL1-A

Parallel Execution among 4 ALUs

$\sum \; j = i+1,\ i+5,\ i+9,\ \cdots,\ L-3$ → IL0-B

$\sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2$ → IL1-B

$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$ → IL1-B

$\sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4$ → IL1-B

Parallel Execution among 4 ALUs

$\sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2$ → IL0-C

$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$ → IL1-C

$\sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4$ → IL1-C

$\sum \; j = i+5,\ i+9,\ i+13,\ \cdots,\ L-3$ → IL1-C

Parallel Execution among 4 ALUs

$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$ → IL0-D

$\sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4$ → IL1-D

$\sum \; j = i+5,\ i+9,\ i+13,\ \cdots,\ L-3$ → IL1-D

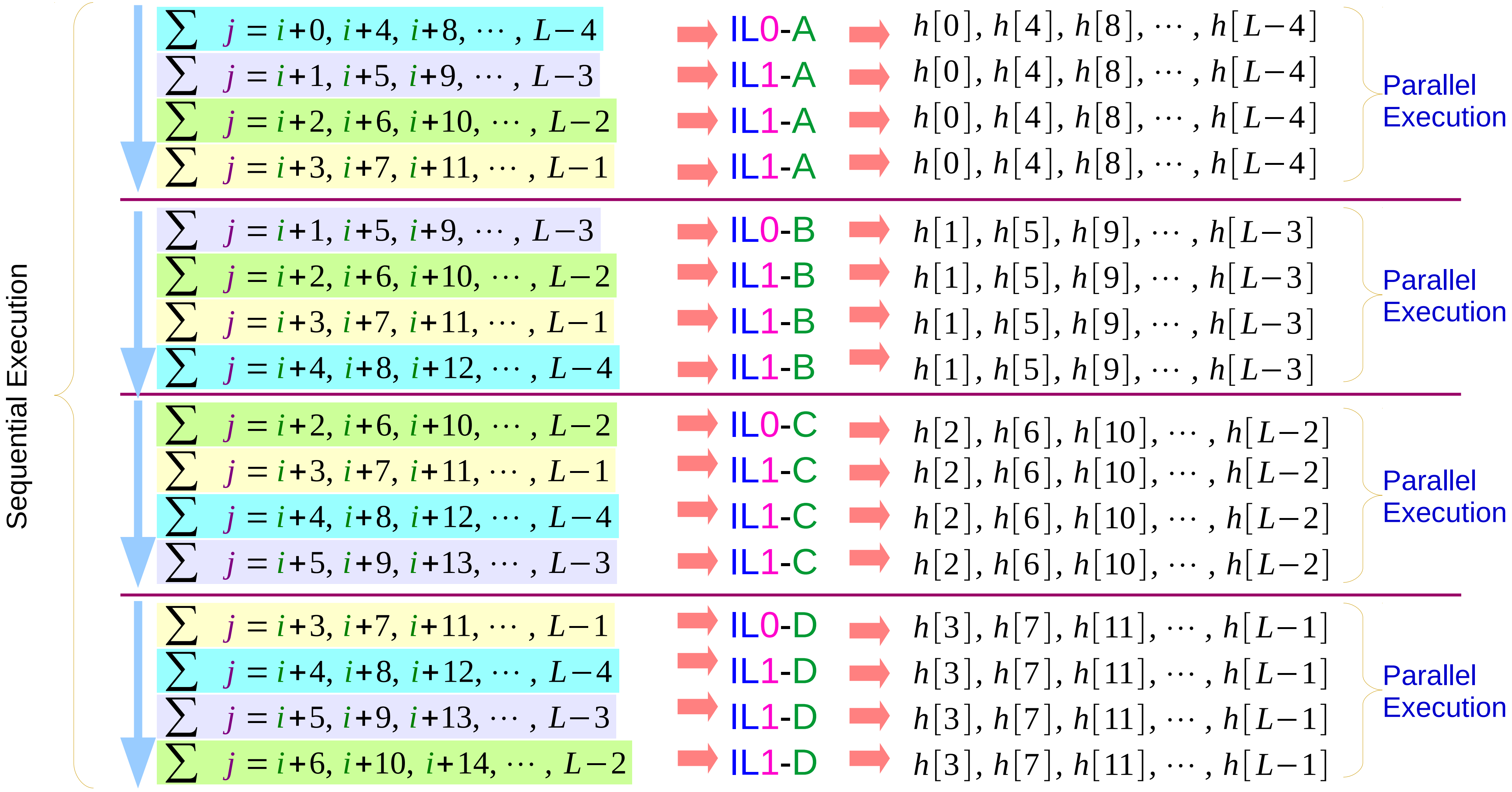
$\sum \; j = i+6,\ i+10,\ i+14,\ \cdots,\ L-2$ → IL1-D

Parallel Execution among 4 ALUs

```
}
```

Memory Access Pattern of h[j]

for (i = L-4; i >= 0; i-=4) {



$\sum \; j = i+0,\ i+4,\ i+8,\ \cdots,\ L-4$	IL0-A	$h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4]$
$\sum \; j = i+1,\ i+5,\ i+9,\ \cdots,\ L-3$	IL1-A	$h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4]$
$\sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2$	IL1-A	$h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4]$
$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$	IL1-A	$h[0],\ h[4],\ h[8],\ \cdots,\ h[L-4]$
$\sum \; j = i+1,\ i+5,\ i+9,\ \cdots,\ L-3$	IL0-B	$h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3]$
$\sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2$	IL1-B	$h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3]$
$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$	IL1-B	$h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3]$
$\sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4$	IL1-B	$h[1],\ h[5],\ h[9],\ \cdots,\ h[L-3]$
$\sum \; j = i+2,\ i+6,\ i+10,\ \cdots,\ L-2$	IL0-C	$h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2]$
$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$	IL1-C	$h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2]$
$\sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4$	IL1-C	$h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2]$
$\sum \; j = i+5,\ i+9,\ i+13,\ \cdots,\ L-3$	IL1-C	$h[2],\ h[6],\ h[10],\ \cdots,\ h[L-2]$
$\sum \; j = i+3,\ i+7,\ i+11,\ \cdots,\ L-1$	IL0-D	$h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1]$
$\sum \; j = i+4,\ i+8,\ i+12,\ \cdots,\ L-4$	IL1-D	$h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1]$
$\sum \; j = i+5,\ i+9,\ i+13,\ \cdots,\ L-3$	IL1-D	$h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1]$
$\sum \; j = i+6,\ i+10,\ i+14,\ \cdots,\ L-2$	IL1-D	$h[3],\ h[7],\ h[11],\ \cdots,\ h[L-1]$

}

Loop Unroll to IL0, IL1, IL2, IL3

```
x_curr = X[i]
h0 = h[0]; h1 = h2 = h3 = 0;
for (j = i; j < L_SUBFR; j+=4)
{
    L_s0 = L_mac(L_s0, x_curr, h0);
    L_s1 = L_mac(L_s1, x_curr, h1);
    L_s2 = L_mac(L_s2, x_curr, h2);
    L_s3 = L_mac(L_s3, x_curr, h3);
    h3 = h[j+1-i]; x_curr = X[j+1];

    L_s0 = L_mac(L_s0, x_curr, h3);
    L_s1 = L_mac(L_s1, x_curr, h0);
    L_s2 = L_mac(L_s2, x_curr, h1);
    L_s3 = L_mac(L_s3, x_curr, h2);
    h2 = h[j+2-i]; x_curr = X[j+2];

    L_s0 = L_mac(L_s0, x_curr, h2);
    L_s1 = L_mac(L_s1, x_curr, h3);
    L_s2 = L_mac(L_s2, x_curr, h0);
    L_s3 = L_mac(L_s3, x_curr, h1);
    h1 = h[j+3-i]; x_curr = X[j+3];

    L_s0 = L_mac(L_s0, x_curr, h1);
    L_s1 = L_mac(L_s1, x_curr, h2);
    L_s2 = L_mac(L_s2, x_curr, h3);
    L_s3 = L_mac(L_s3, x_curr, h0);
    h0 = h[j+4-i]; x_curr = X[j+4];
}
```

```
x_curr = X[i]
h0 = h[0]; h1 = h2 = h3 = 0;
for (j = i; j < L_SUBFR; j+=4)
{
    L_s0 = L_mac(L_s0, x_curr, h0);    →  IL0-A
    L_s1 = L_mac(L_s1, x_curr, h1);    →  IL1-A
    L_s2 = L_mac(L_s2, x_curr, h2);    →  IL2-A
    L_s3 = L_mac(L_s3, x_curr, h3);    →  IL3-A
    h3 = h[j+1-i]; x_curr = X[j+1];    ←  Update

    L_s0 = L_mac(L_s0, x_curr, h3);    →  IL0-B
    L_s1 = L_mac(L_s1, x_curr, h0);    →  IL1-B
    L_s2 = L_mac(L_s2, x_curr, h1);    →  IL2-B
    L_s3 = L_mac(L_s3, x_curr, h2);    →  IL3-B
    h2 = h[j+2-i]; x_curr = X[j+2];    ←  Update

    L_s0 = L_mac(L_s0, x_curr, h2);    →  IL0-C
    L_s1 = L_mac(L_s1, x_curr, h3);    →  IL1-C
    L_s2 = L_mac(L_s2, x_curr, h0);    →  IL2-C
    L_s3 = L_mac(L_s3, x_curr, h1);    →  IL3-C
    h1 = h[j+3-i]; x_curr = X[j+3];    ←  Update

    L_s0 = L_mac(L_s0, x_curr, h1);    →  IL0-D
    L_s1 = L_mac(L_s1, x_curr, h2);    →  IL1-D
    L_s2 = L_mac(L_s2, x_curr, h3);    →  IL2-D
    L_s3 = L_mac(L_s3, x_curr, h0);    →  IL3-D
    h0 = h[j+4-i]; x_curr = X[j+4];    ←  Update
}
```

Rearranged for easy understanding

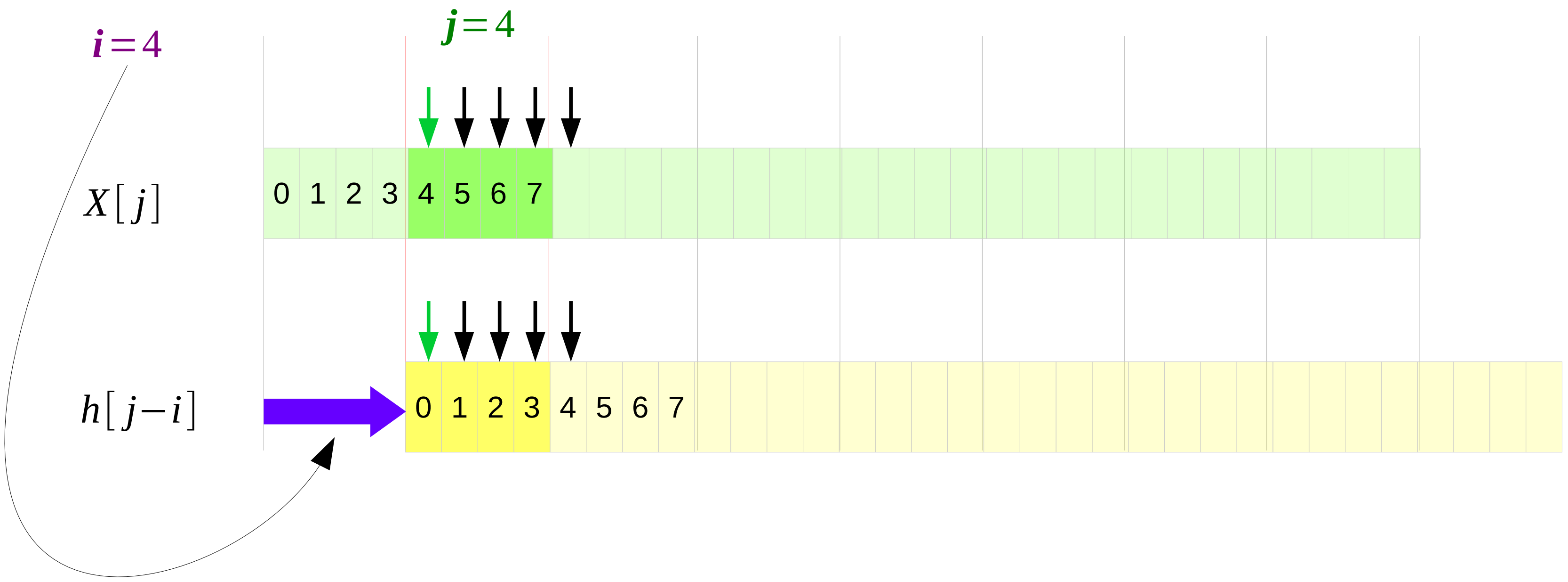
```
{
    L_s0 = L_mac(L_s0, x_curr, h0);   h3 = h[j+1-i]; x_curr = X[j+1];  ➡  IL0-A  ➡  L_s0 += X[j+0] * h[j+0-i];
    L_s0 = L_mac(L_s0, x_curr, h3);   h2 = h[j+2-i]; x_curr = X[j+2];  ➡  IL0-B  ➡  L_s0 += X[j+1] * h[j+1-i];
    L_s0 = L_mac(L_s0, x_curr, h2);   h1 = h[j+3-i]; x_curr = X[j+3];  ➡  IL0-C  ➡  L_s0 += X[j+2] * h[j+2-i];
    L_s0 = L_mac(L_s0, x_curr, h1);   h0 = h[j+4-i]; x_curr = X[j+4];  ➡  IL0-D  ➡  L_s0 += X[j+3] * h[j+3-i];
}

{
    L_s1 = L_mac(L_s1, x_curr, h1);   h3 = h[j+1-i]; x_curr = X[j+1];  ➡  IL1-A  ➡  L_s1 += X[j+0] * h[j+3-i];
    L_s1 = L_mac(L_s1, x_curr, h0);   h2 = h[j+2-i]; x_curr = X[j+2];  ➡  IL1-B  ➡  L_s1 += X[j+1] * h[j+0-i];
    L_s1 = L_mac(L_s1, x_curr, h3);   h1 = h[j+3-i]; x_curr = X[j+3];  ➡  IL1-C  ➡  L_s1 += X[j+2] * h[j+1-i];
    L_s1 = L_mac(L_s1, x_curr, h2);   h0 = h[j+4-i]; x_curr = X[j+4];  ➡  IL1-D  ➡  L_s1 += X[j+3] * h[j+2-i];
}

{
    L_s2 = L_mac(L_s2, x_curr, h2);   h3 = h[j+1-i]; x_curr = X[j+1];  ➡  IL2-A  ➡  L_s2 += X[j+0] * h[j+2-i];
    L_s2 = L_mac(L_s2, x_curr, h1);   h2 = h[j+2-i]; x_curr = X[j+2];  ➡  IL2-B  ➡  L_s2 += X[j+1] * h[j+3-i];
    L_s2 = L_mac(L_s2, x_curr, h0);   h1 = h[j+3-i]; x_curr = X[j+3];  ➡  IL2-C  ➡  L_s2 += X[j+2] * h[j+0-i];
    L_s2 = L_mac(L_s2, x_curr, h3);   h0 = h[j+4-i]; x_curr = X[j+4];  ➡  IL2-C  ➡  L_s2 += X[j+3] * h[j+1-i];
}

{
    L_s3 = L_mac(L_s3, x_curr, h3);   h3 = h[j+1-i]; x_curr = X[j+1];  ➡  IL3-A  ➡  L_s3 += X[j+0] * h[j+1-i];
    L_s3 = L_mac(L_s3, x_curr, h2);   h2 = h[j+2-i]; x_curr = X[j+2];  ➡  IL3-B  ➡  L_s3 += X[j+1] * h[j+2-i];
    L_s3 = L_mac(L_s3, x_curr, h1);   h1 = h[j+3-i]; x_curr = X[j+3];  ➡  IL3-C  ➡  L_s3 += X[j+2] * h[j+3-i];
    L_s3 = L_mac(L_s3, x_curr, h0);   h0 = h[j+4-i]; x_curr = X[j+4];  ➡  IL3-D  ➡  L_s3 += X[j+3] * h[j+0-i];
}
```

Change h3 ↔ h1

Updating h0, h1, h2, h3

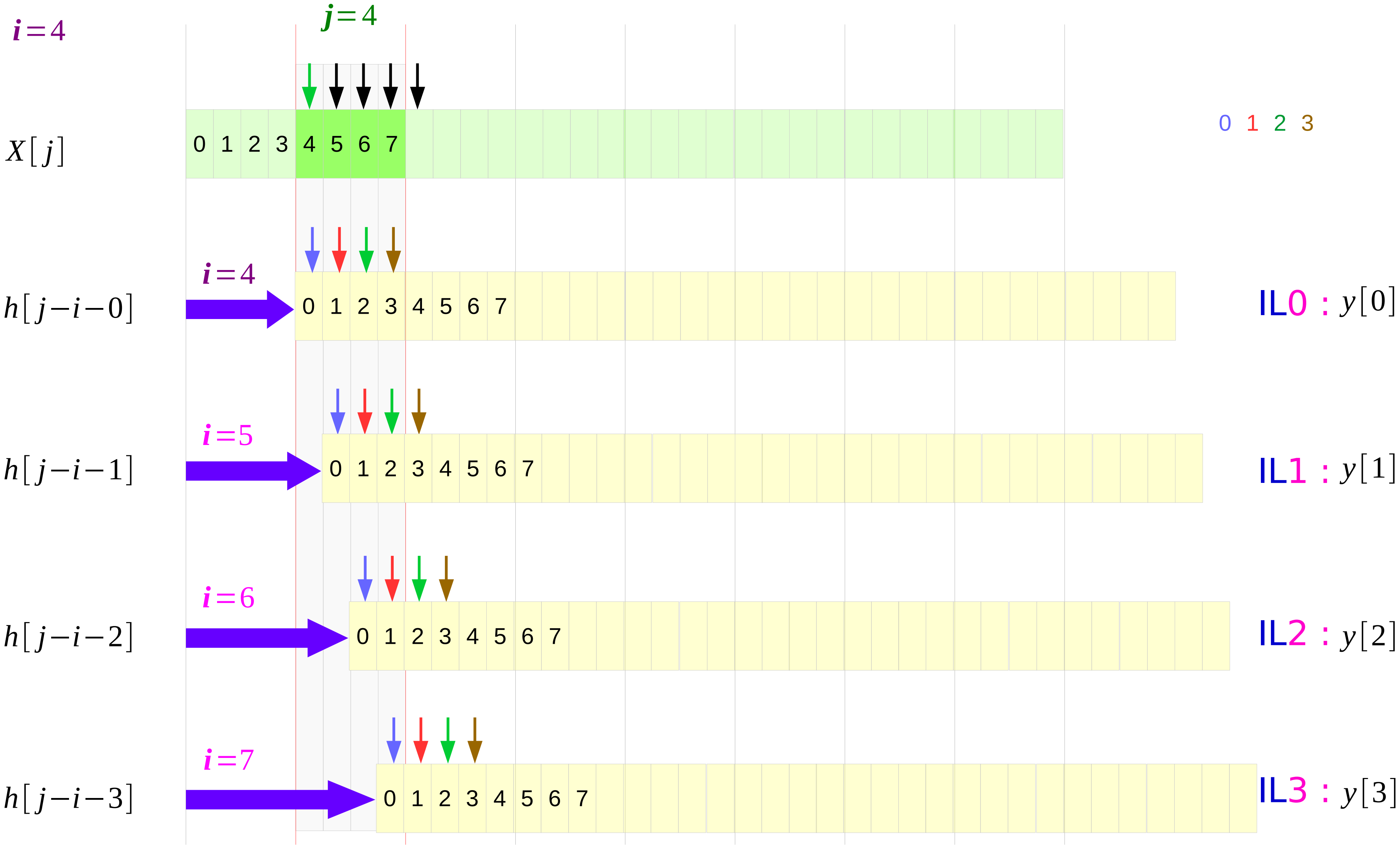
```
{                                 old
    L_s0 = L_mac(L_s0, x_curr, h0);    h1 = h[j+1-i]; x_curr = X[j+1];        L_s0 += X[j+0] * h[j+0-i];   old
    L_s0 = L_mac(L_s0, x_curr, h1);    h2 = h[j+2-i]; x_curr = X[j+2];        L_s0 += X[j+1] * h[j+1-i];
    L_s0 = L_mac(L_s0, x_curr, h2);    h3 = h[j+3-i]; x_curr = X[j+3];        L_s0 += X[j+2] * h[j+2-i];
    L_s0 = L_mac(L_s0, x_curr, h3);    h0 = h[j+4-i]; x_curr = X[j+4];        L_s0 += X[j+3] * h[j+3-i];
}

{                                 old
    L_s1 = L_mac(L_s1, x_curr, h3);    h1 = h[j+1-i]; x_curr = X[j+1];        L_s1 += X[j+0] * h[j+3-i];   old
    L_s1 = L_mac(L_s1, x_curr, h0);    h2 = h[j+2-i]; x_curr = X[j+2];        L_s1 += X[j+1] * h[j+0-i];
    L_s1 = L_mac(L_s1, x_curr, h1);    h3 = h[j+3-i]; x_curr = X[j+3];        L_s1 += X[j+2] * h[j+1-i];
    L_s1 = L_mac(L_s1, x_curr, h2);    h0 = h[j+4-i]; x_curr = X[j+4];        L_s1 += X[j+3] * h[j+2-i];
}

{                                 old
    L_s2 = L_mac(L_s2, x_curr, h2);    h1 = h[j+1-i]; x_curr = X[j+1];        L_s2 += X[j+0] * h[j+2-i];   old
    L_s2 = L_mac(L_s2, x_curr, h3);    h2 = h[j+2-i]; x_curr = X[j+2];        L_s2 += X[j+1] * h[j+3-i];
    L_s2 = L_mac(L_s2, x_curr, h0);    h3 = h[j+3-i]; x_curr = X[j+3];        L_s2 += X[j+2] * h[j+0-i];
    L_s2 = L_mac(L_s2, x_curr, h1);    h0 = h[j+4-i]; x_curr = X[j+4];        L_s2 += X[j+3] * h[j+1-i];
}

{                                 old
    L_s3 = L_mac(L_s3, x_curr, h1);    h1 = h[j+1-i]; x_curr = X[j+1];        L_s3 += X[j+0] * h[j+1-i];   old
    L_s3 = L_mac(L_s3, x_curr, h2);    h2 = h[j+2-i]; x_curr = X[j+2];        L_s3 += X[j+1] * h[j+2-i];
    L_s3 = L_mac(L_s3, x_curr, h3);    h3 = h[j+3-i]; x_curr = X[j+3];        L_s3 += X[j+2] * h[j+3-i];
    L_s3 = L_mac(L_s3, x_curr, h0);    h0 = h[j+4-i]; x_curr = X[j+4];        L_s3 += X[j+3] * h[j+0-i];
}
```

Accessing X[j] & h[j-i]

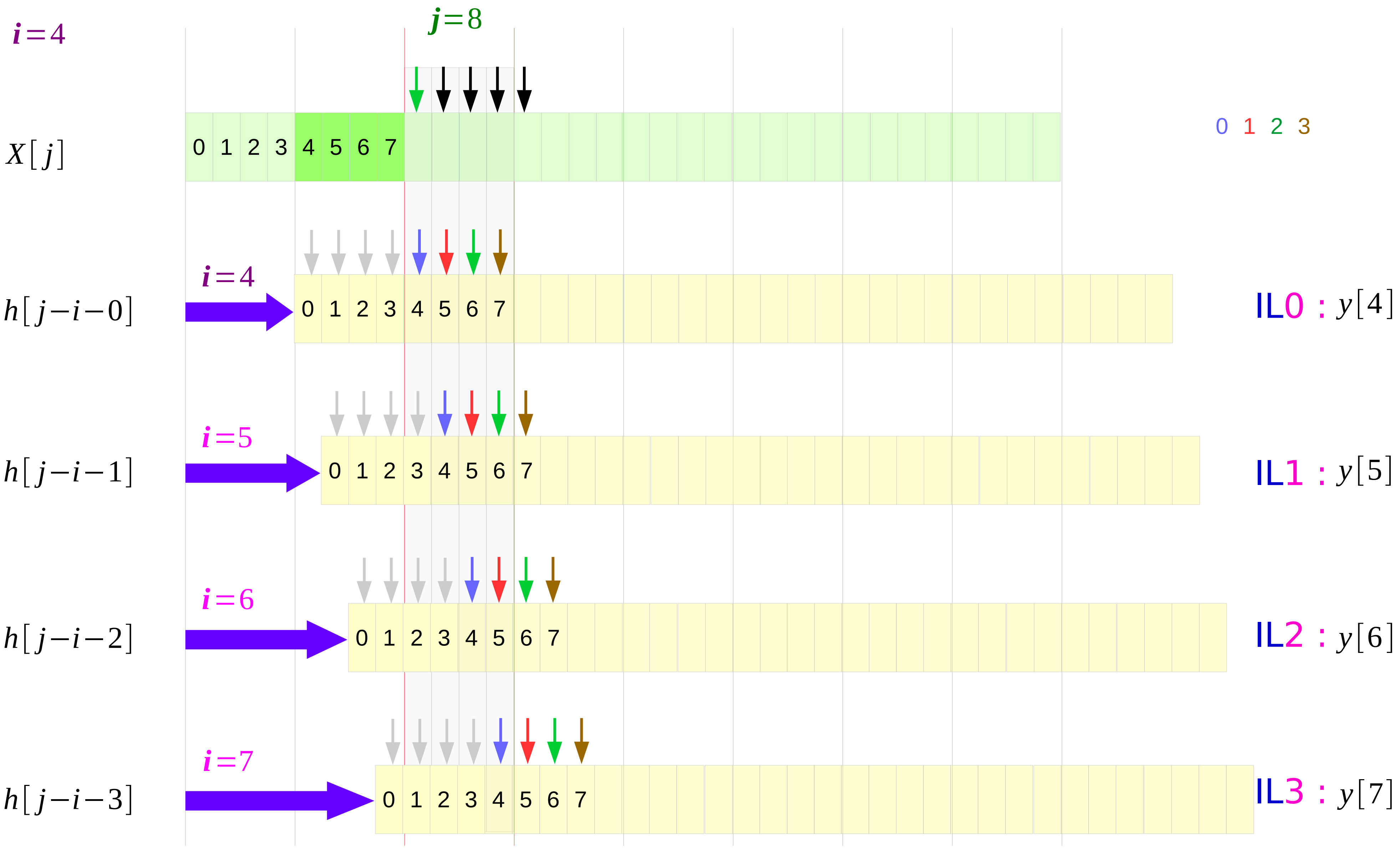


```
h1 = h[j+1-i] = h[j-i+1];      x_curr = X[j+1];
h2 = h[j+2-i] = h[j-i+2];      x_curr = X[j+2];
h3 = h[j+3-i] = h[j-i+3];      x_curr = X[j+3];
h0 = h[j+4-i] = h[j-i+4];      x_curr = X[j+4];
```

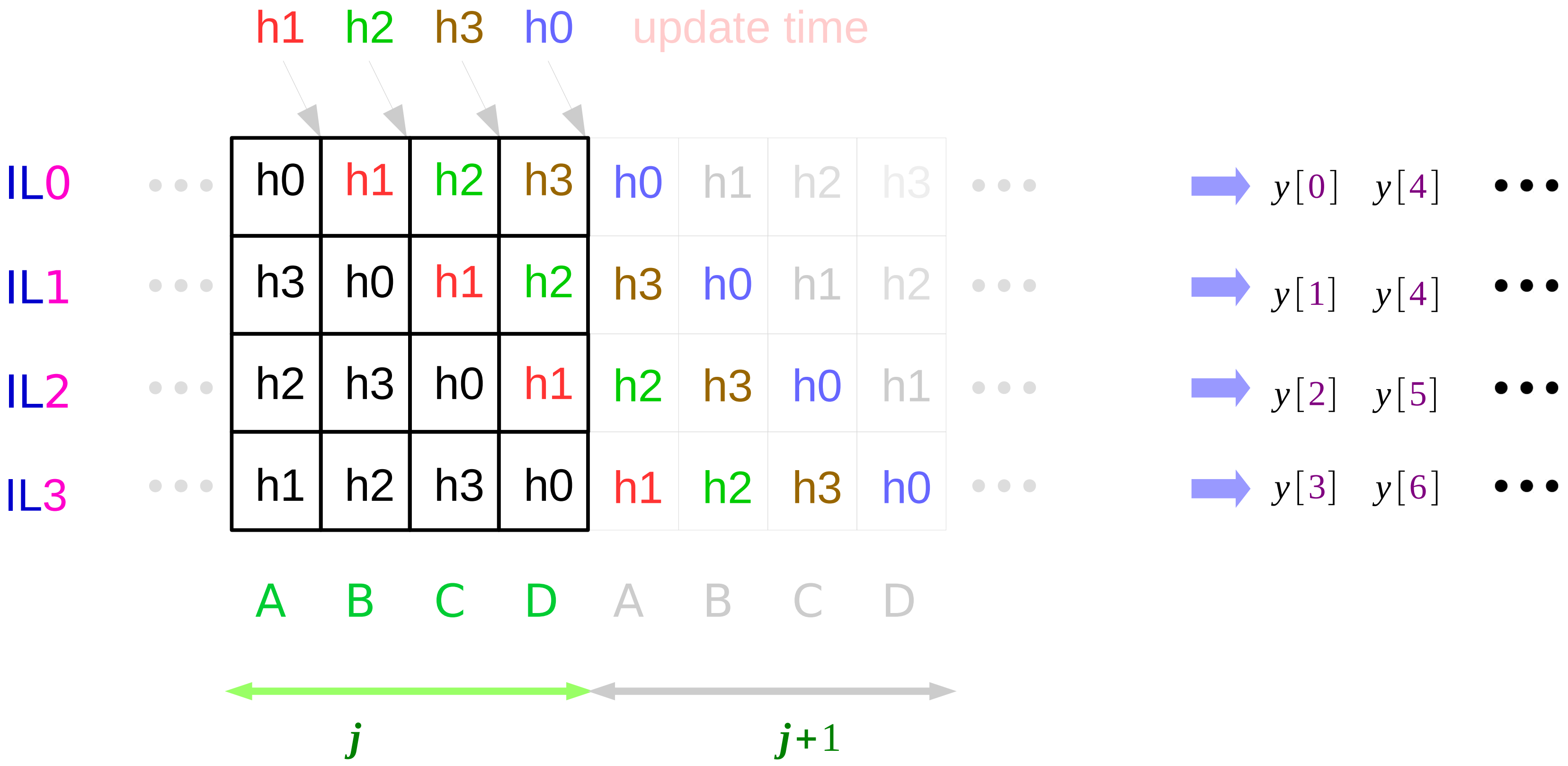
Accessing X[j] & h[j-i]



Accessing X[j] & h[j-i]

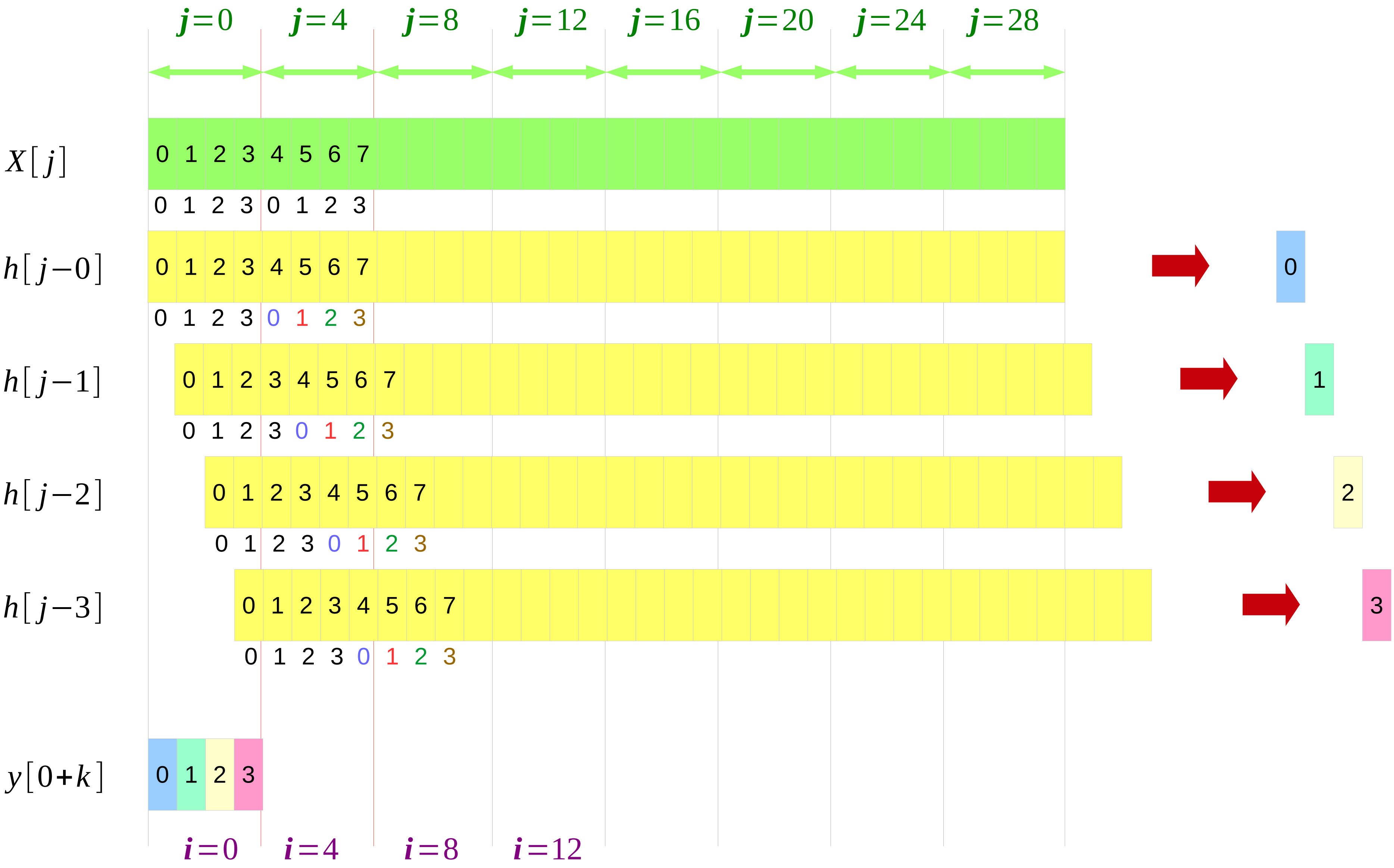


Data Access Patterns: h0, h1, h2, h3

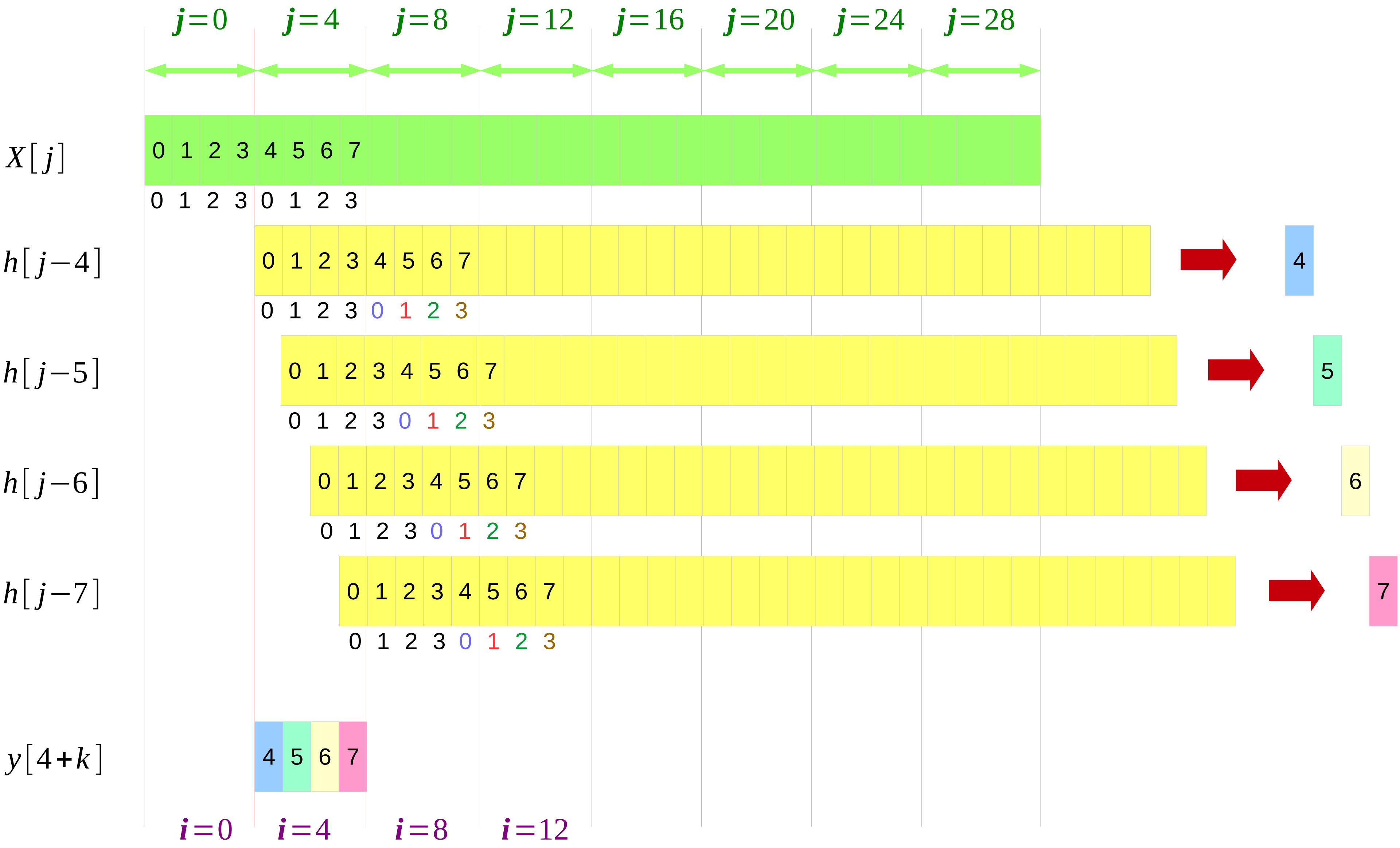


		A	B	C	D	A	B	C	D				
IL0	• • •	h0	h1	h2	h3	h0	h1	h2	h3	• • •	$y[0]$	$y[4]$	• • •
IL1	• • •	h3	h0	h1	h2	h3	h0	h1	h2	• • •	$y[1]$	$y[4]$	• • •
IL2	• • •	h2	h3	h0	h1	h2	h3	h0	h1	• • •	$y[2]$	$y[5]$	• • •
IL3	• • •	h1	h2	h3	h0	h1	h2	h3	h0	• • •	$y[3]$	$y[6]$	• • •

Computing y[0], y[1], y[2], y[3]



Computing y[4], y[5], y[6], y[7]



References

[1] http://www.isis.vanderbilt.edu/akos/eece6354
[2] http://eecs.vanderbilt.edu/courses/ee276/Fall06_lectures/10%20RTOS%20basics.pdf
[3] https://doc.micrium.com/display/osiidoc/home
[4] http://ftp1.digi.com/support/documentation/0220047_e.pdf
[5] http://people.cst.cmich.edu/yelam1k/asee/proceedings/2012/Full%20Papers/Jochum.pdf