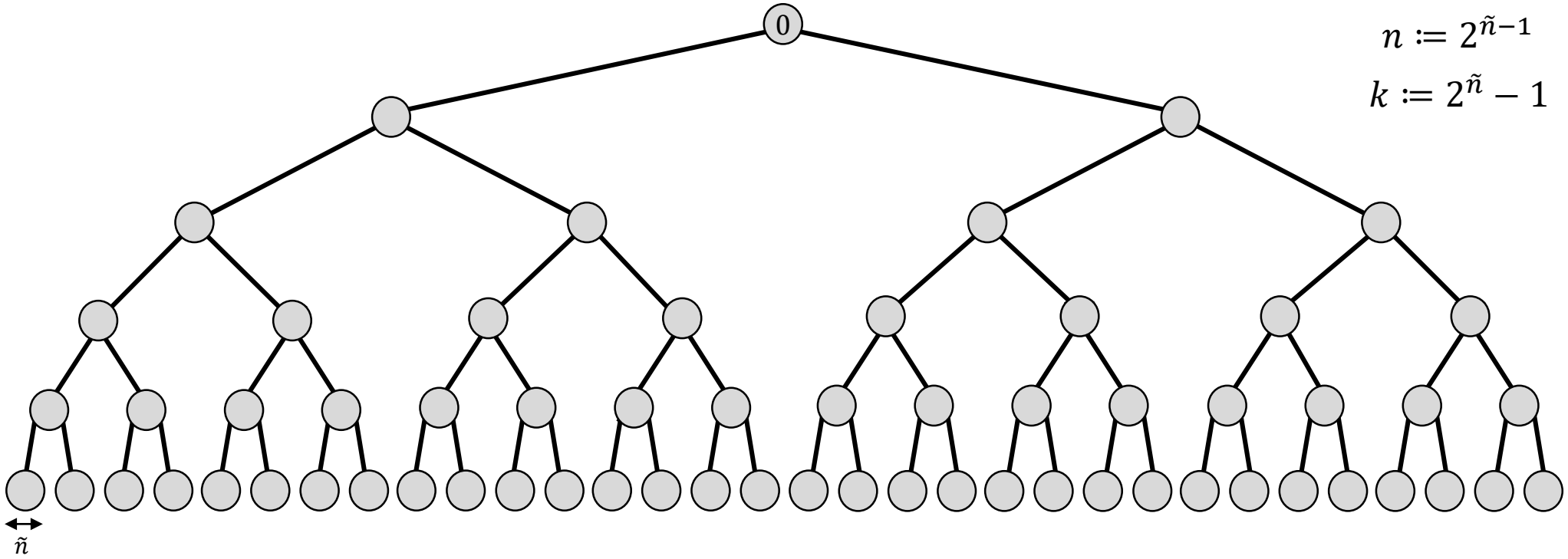
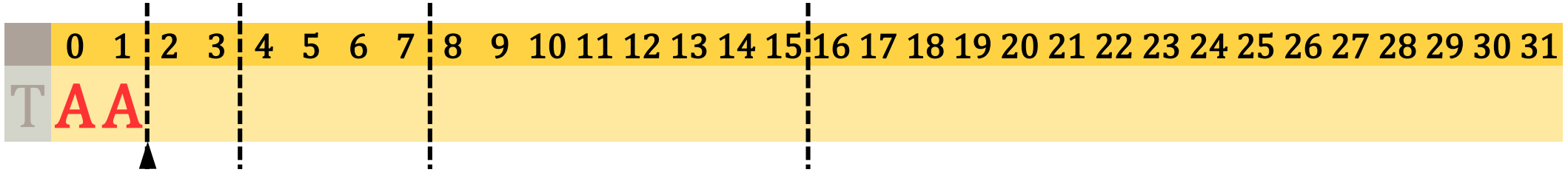
Verwaltung des Arbeitsspeichers

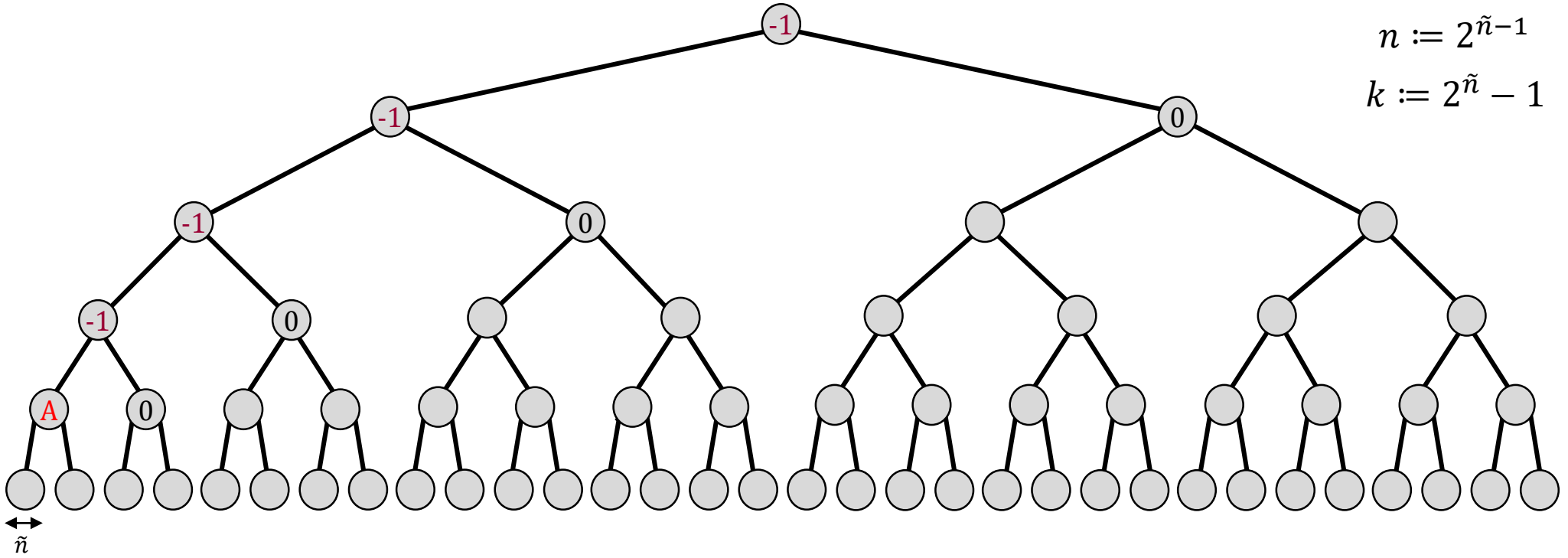
Buddy-Verfahren (Ausgangszustand)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
T																																



Buddy-Verfahren (A Reservieren)

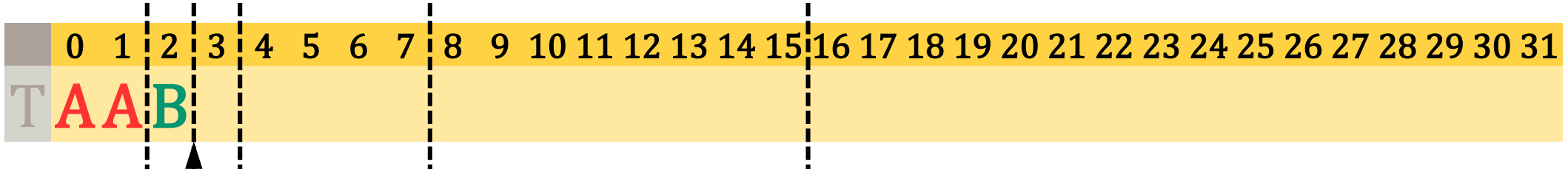


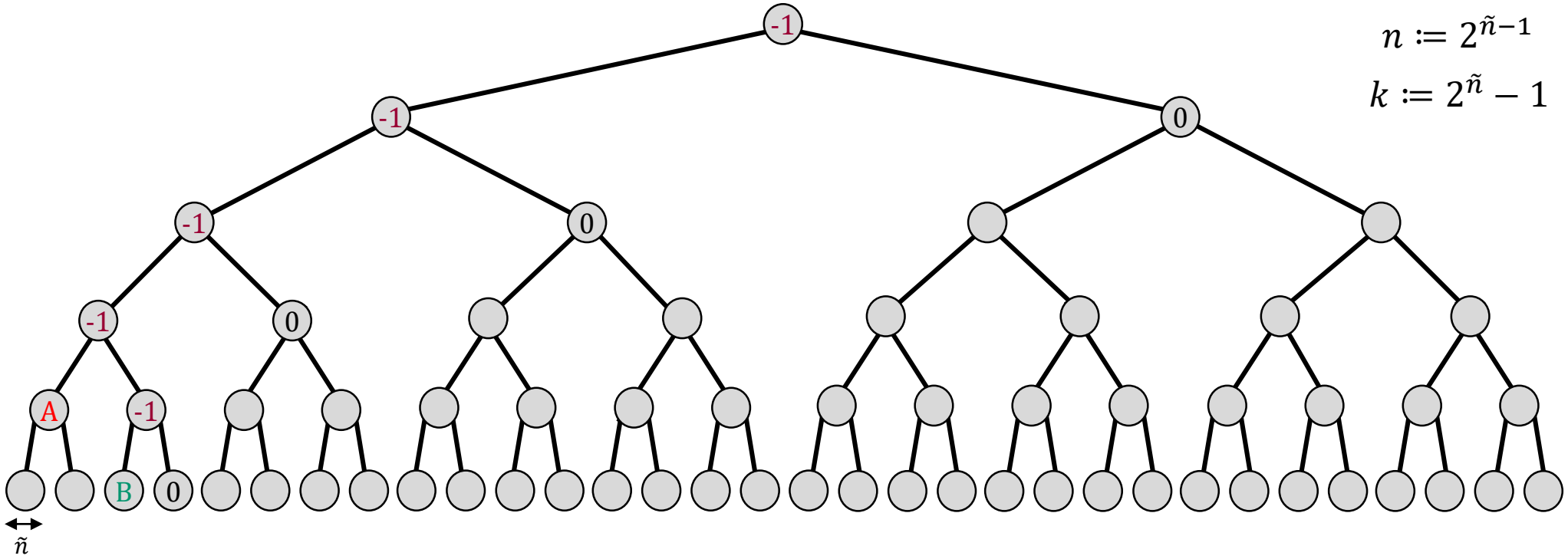


$n := 2^{\tilde{n}-1}$

$k := 2^{\tilde{n}} - 1$

Buddy-Verfahren (B Reservieren)

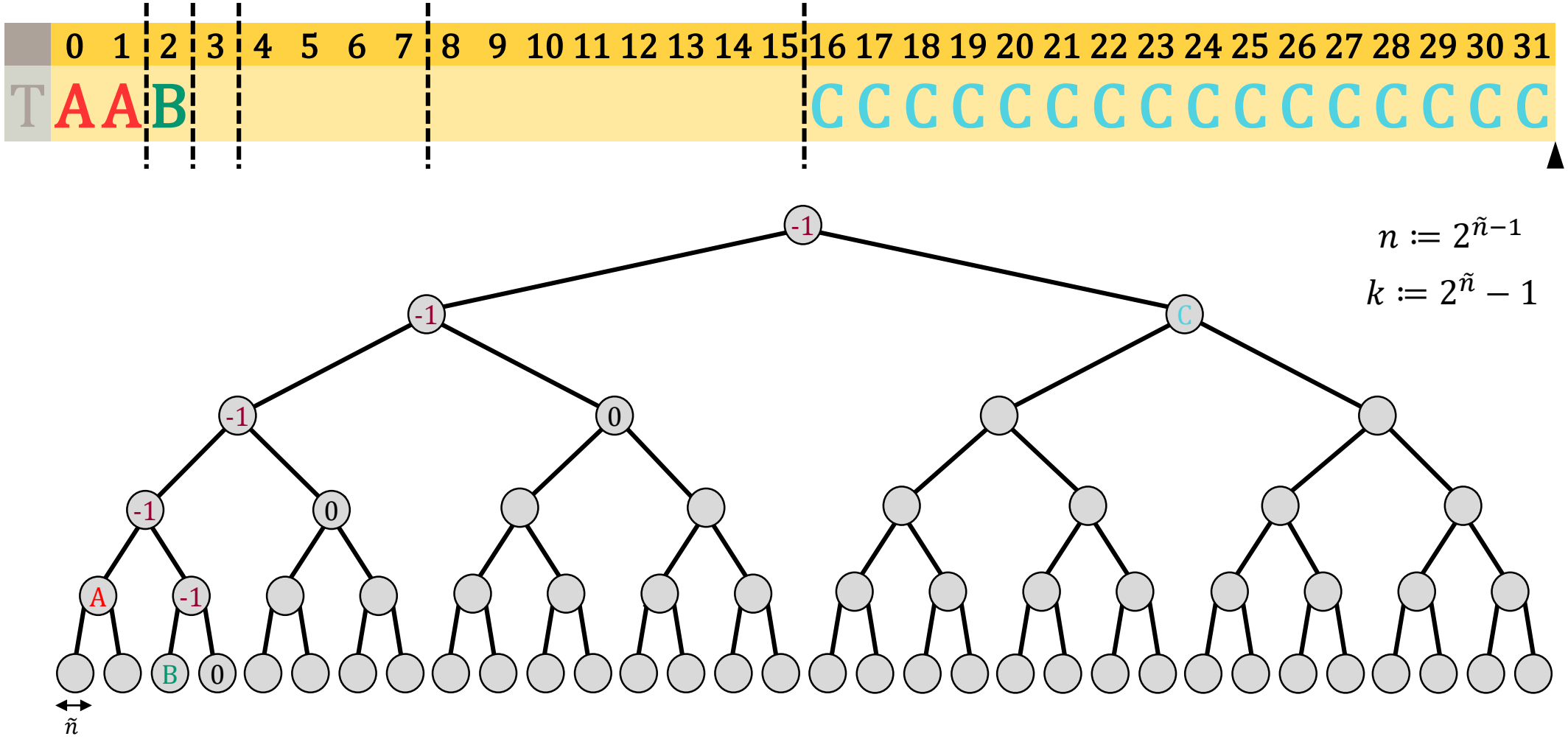




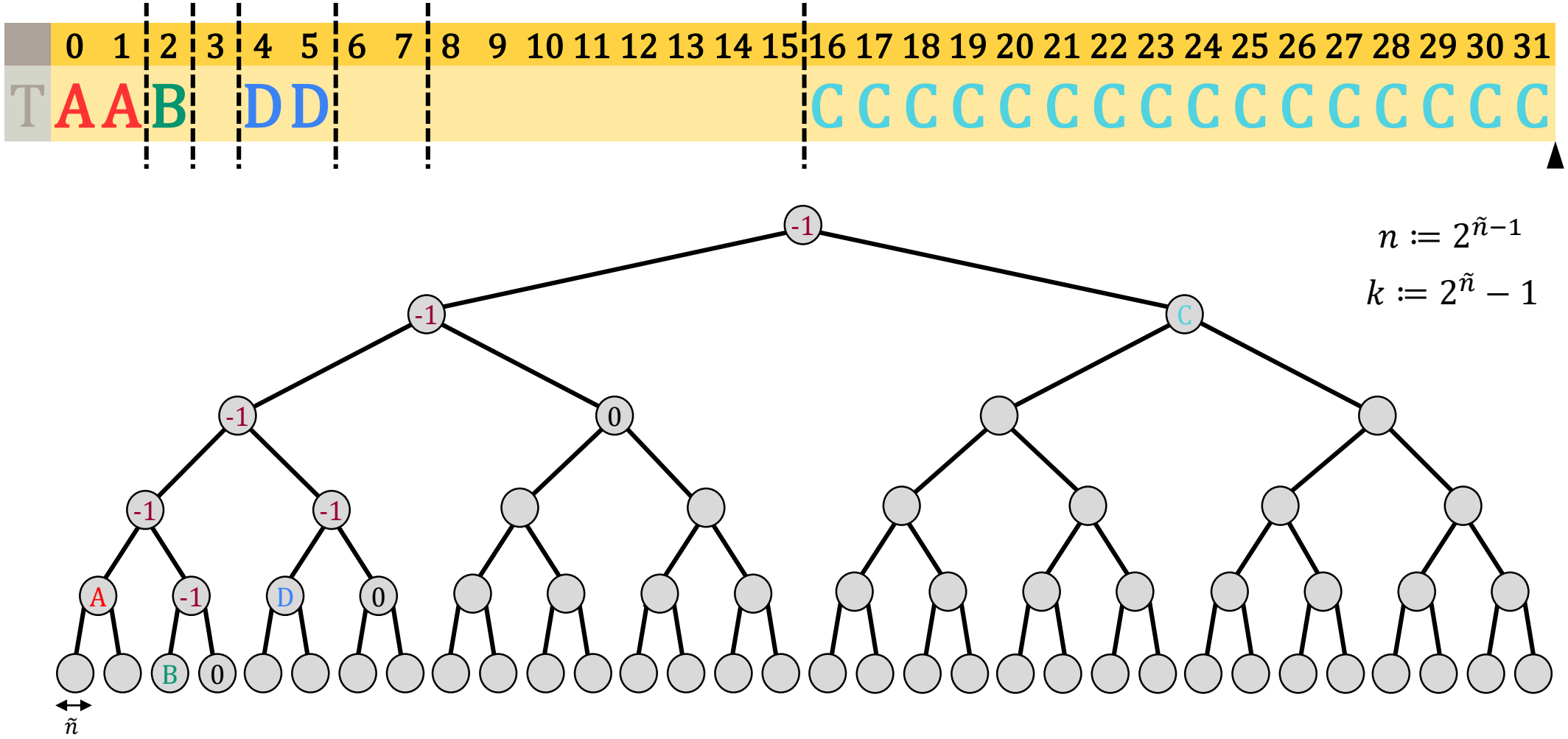
$$n := 2^{\tilde{n}-1}$$

$$k := 2^{\tilde{n}} - 1$$

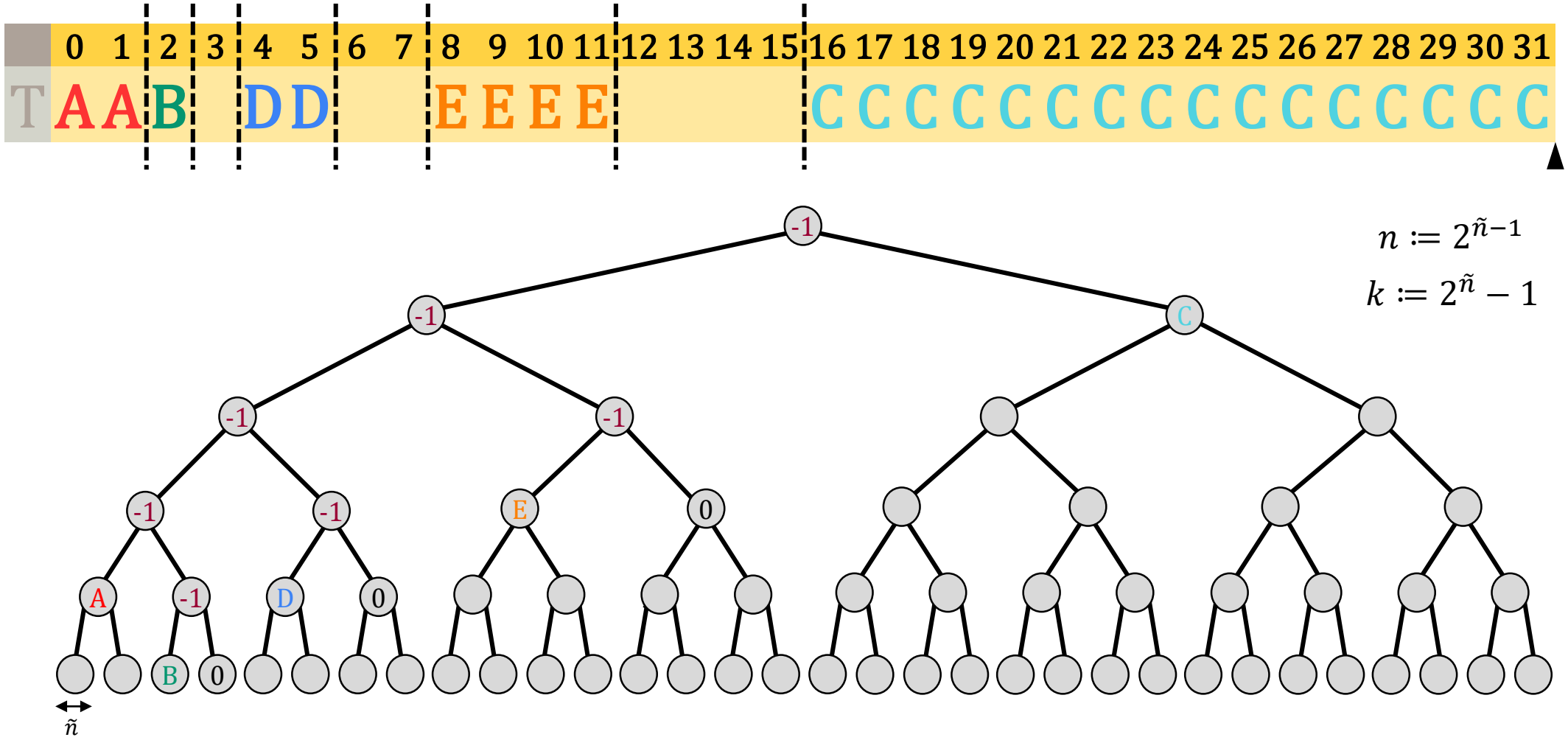
Buddy-Verfahren (C Reservieren)



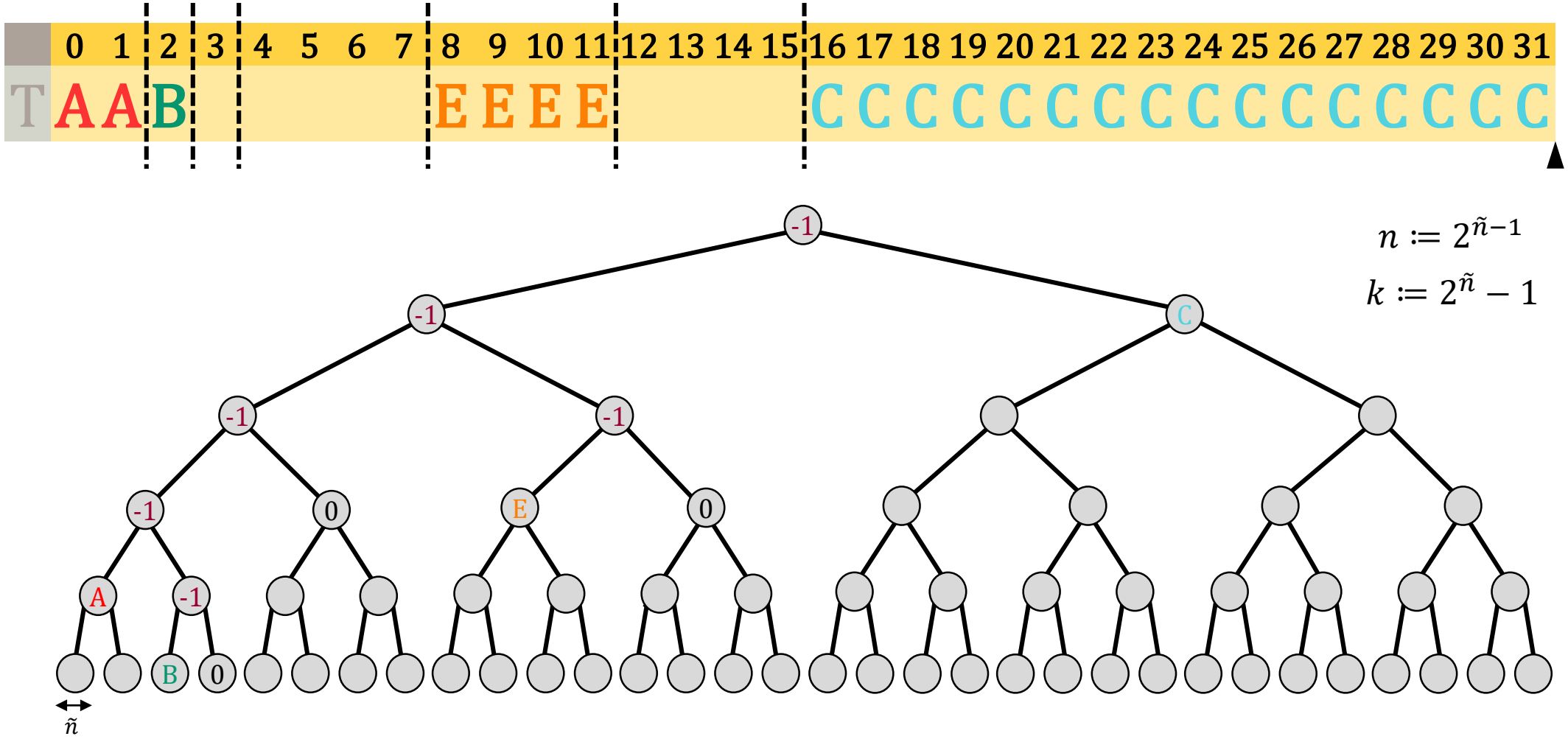
Buddy-Verfahren (D Reservieren)



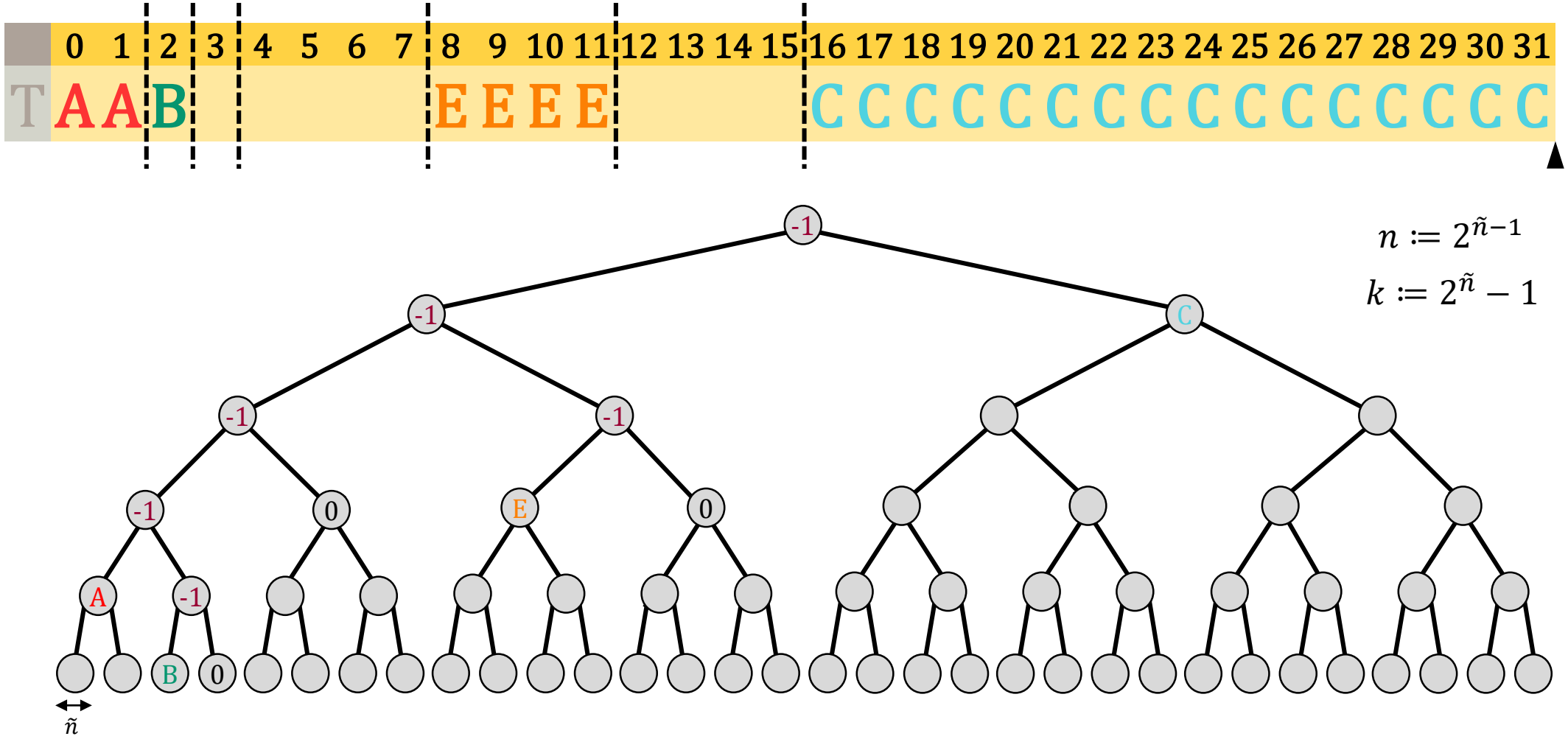
Buddy-Verfahren (E Reservieren)



Buddy-Verfahren (D Freigeben)

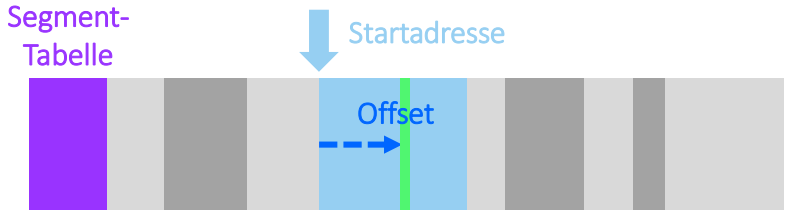


Buddy-Verfahren (F Ablehnen)

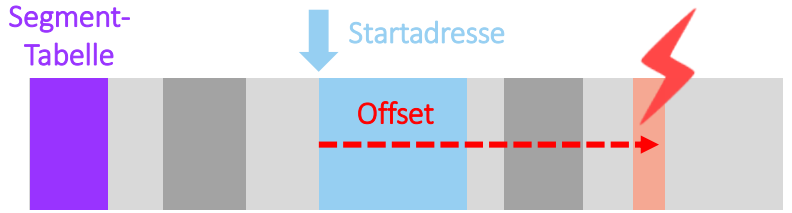


Segmentbasierte Adressberechnung

- Die physikalische Adresse erhält man bei der **segmentbasierten Adressberechnung** durch die Addition von Startadresse mit dem in der virtuellen Adresse enthaltenen Offset (Byteposition in der Seite)
- Ist der Offset größer als die **Segmentlänge** entsteht eine **Speicherschutzverletzung** (engl. **Segmentation-Fault**):

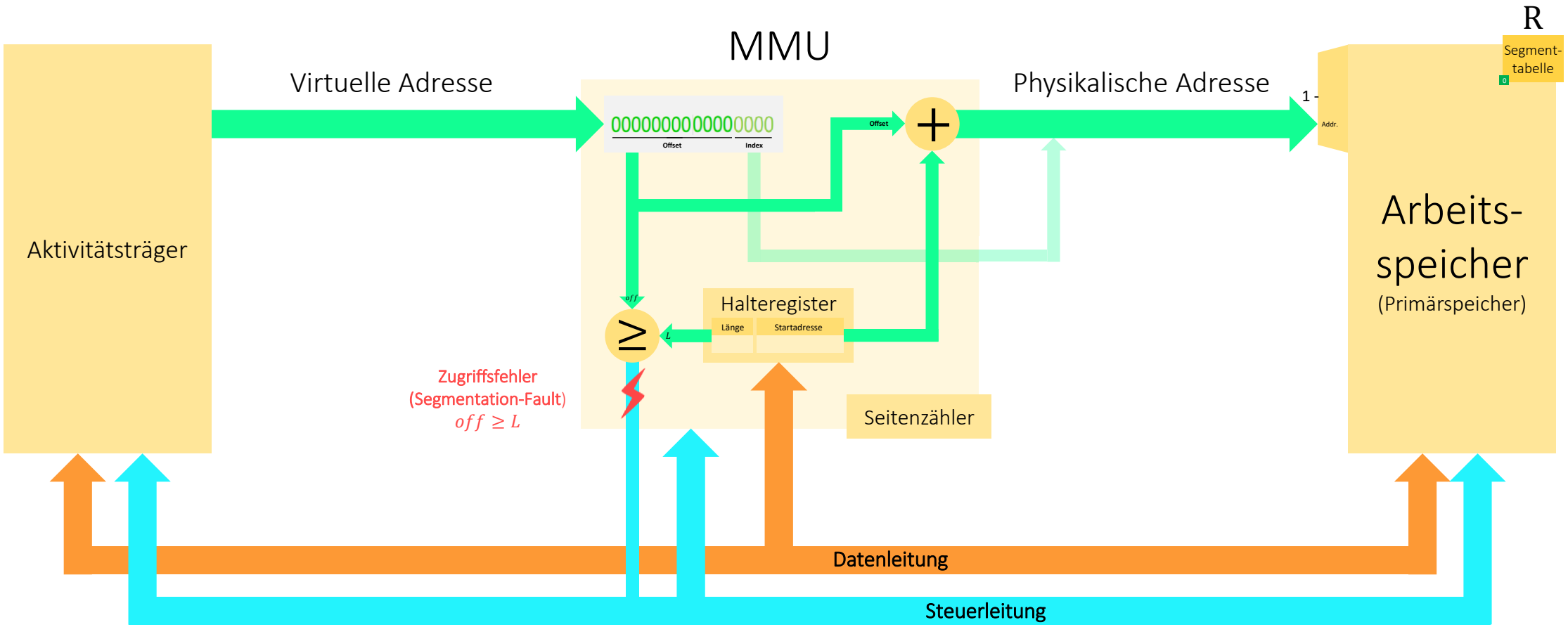


Speicherschutzverletzung:
Offset < Länge



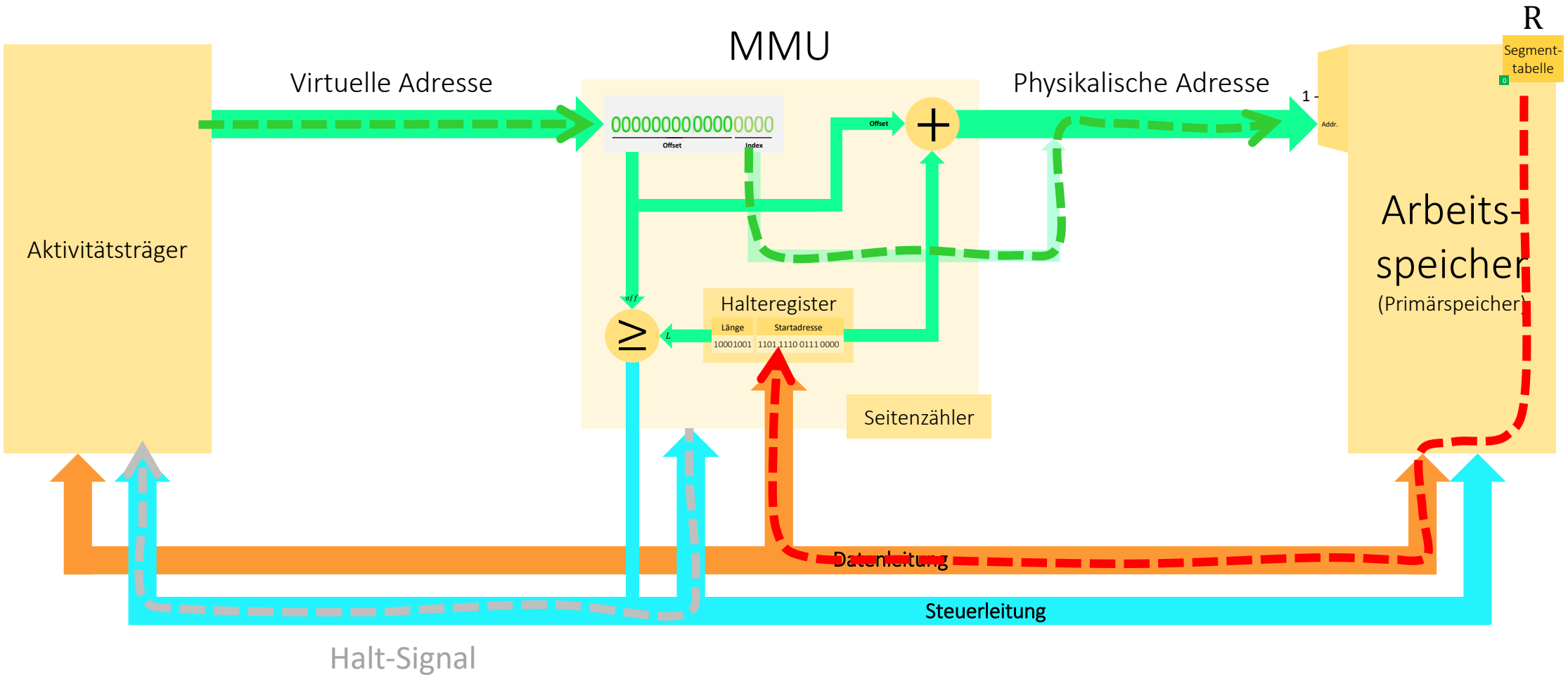
Speicherschutzverletzung:
Offset ≥ Länge

Segmentierung

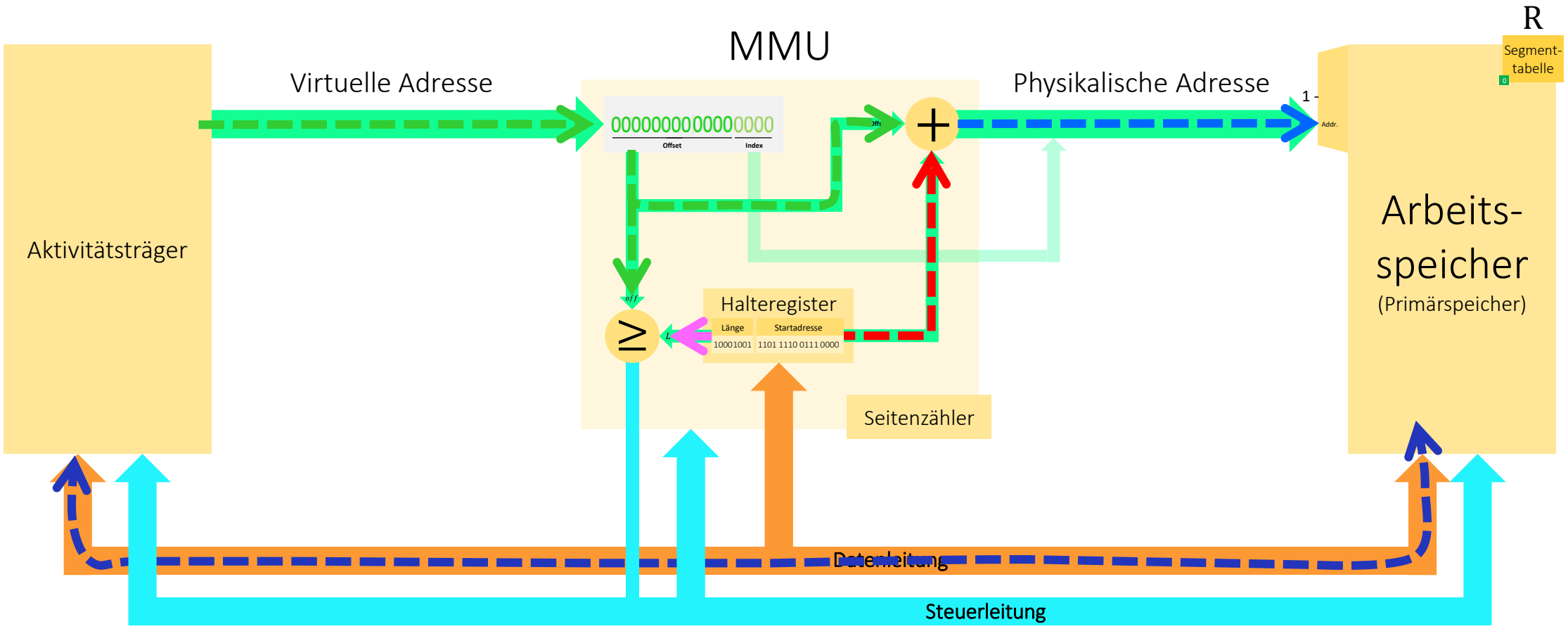


Setzt man $\overrightarrow{\mathbf{Index}} = \vec{0}$, führt die MMU eine Identitätsabbildung aus: So kann direkt auf Segmenttabelle $\mathbf{R}$ zugegriffen werden.

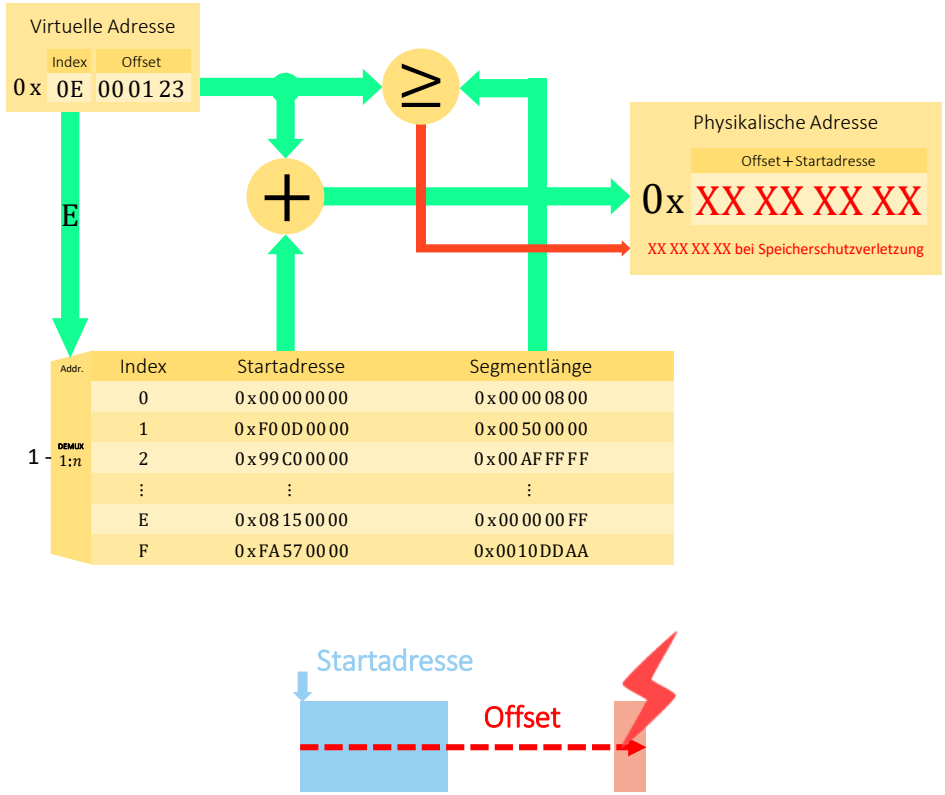
Holen der Segmentnummer (Falls vorhanden)



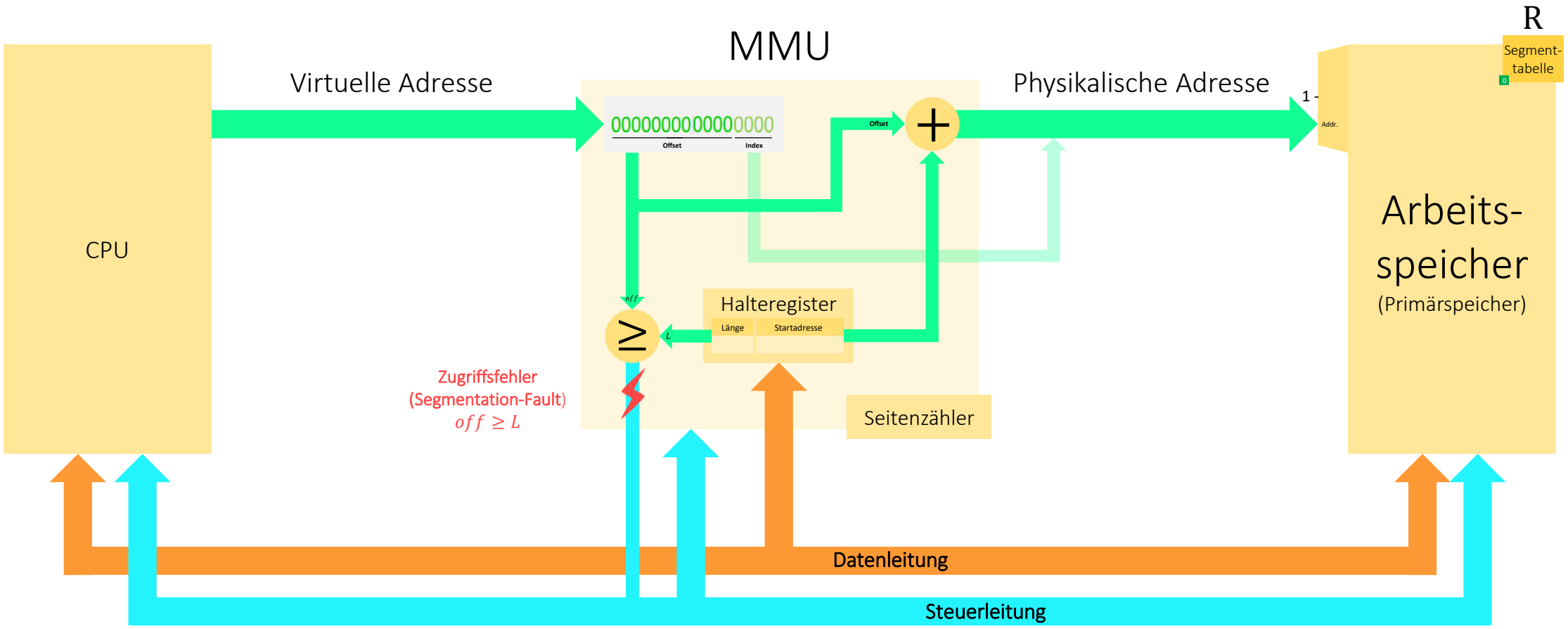
Zusammensetzen der physikalischen Adresse II



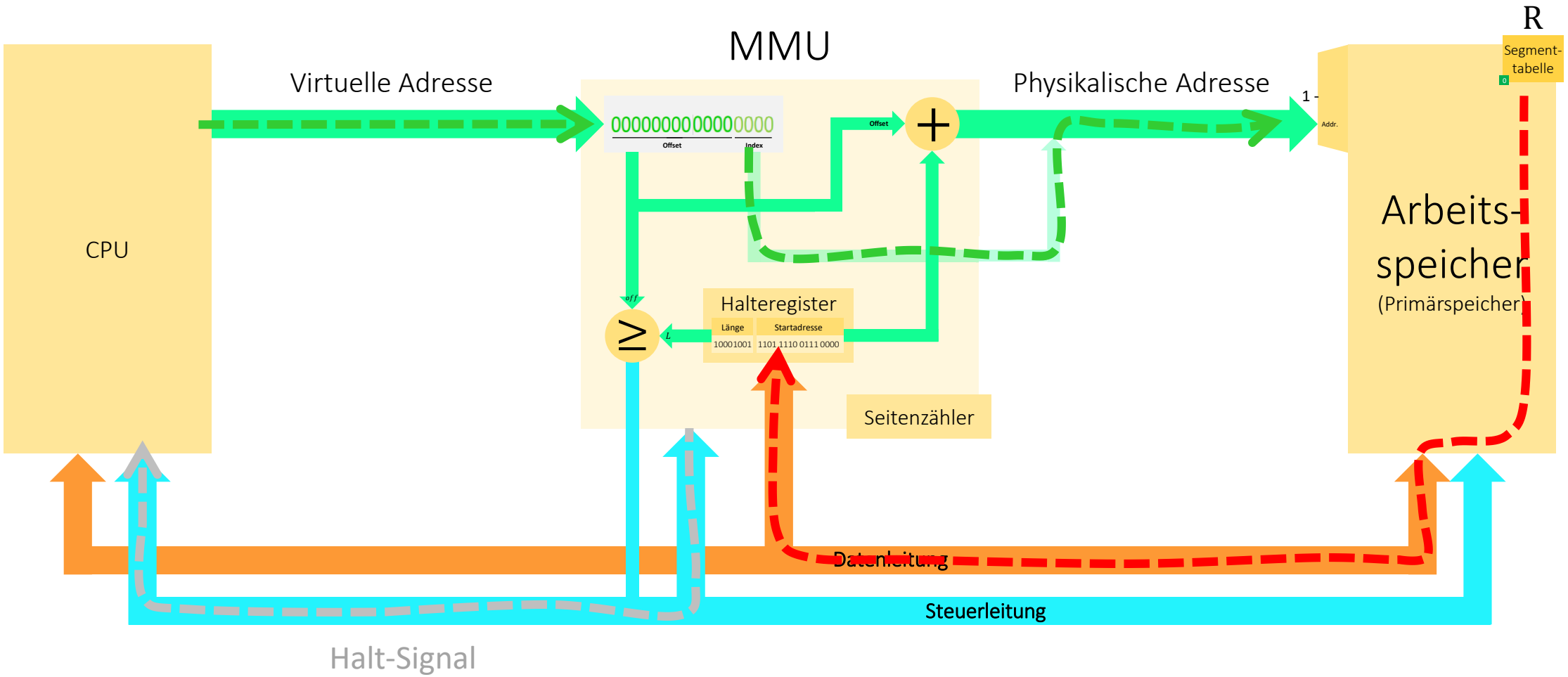
Segmentbasierte Adressberechnung (Beispiel)



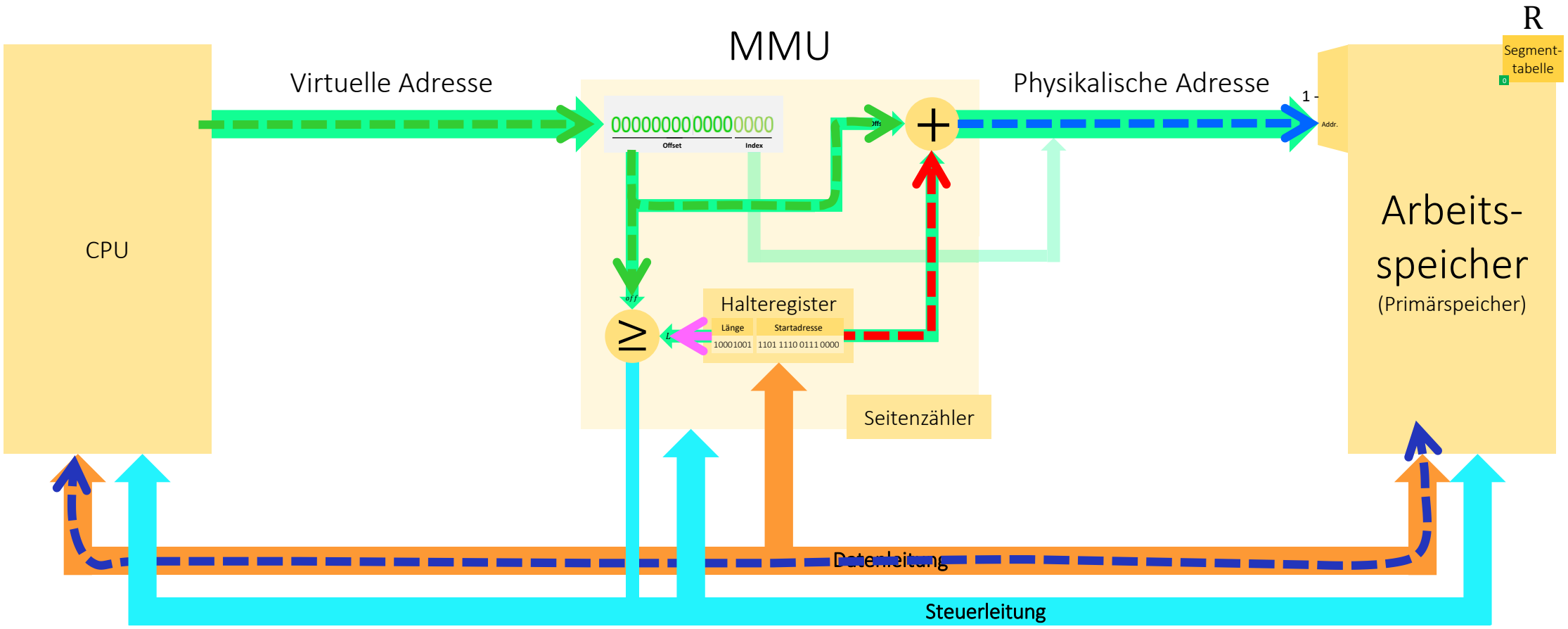
Segmentierung



Holen der Segmentnummer (Falls vorhanden)



Zusammensetzen der physikalischen Adresse II



LRU-Algorithmus

Beispiel:

Registertabelle

Zugriff:	1	5	3	3	5	4	4	2	7	4	9	1	4	6
Kachel 0:	1	1	1	1	1	1	1	2	2	2	2	1	1	1
Kachel 2:	-	5	5	5	5	5	5	5	5	5	9	9	9	9
Kachel 2:	-	-	3	3	3	3	3	3	7	7	7	7	7	6
Kachel 3:	-	-	-	-	-	4	4	4	4	4	4	4	4	4

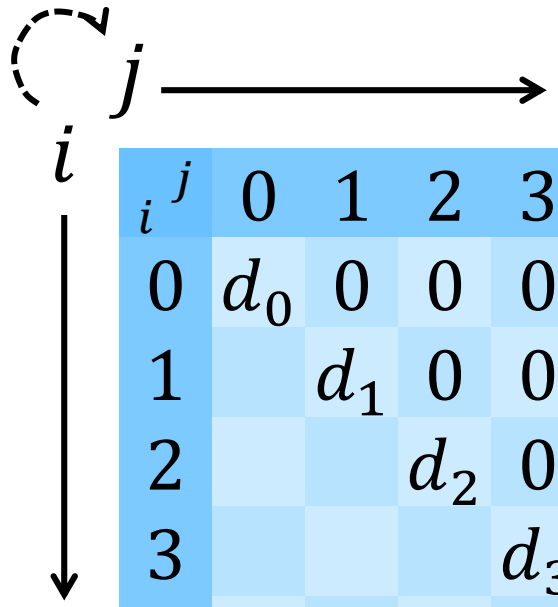
Zugriffstabelle (Zahl der Zyklen seit des letzten Aufrufs)

Kachel 0:	0	1	2	3	4	5	6	0	1	2	3	0	1	2
Kachel 2:	INT_MAX	0	1	2	0	1	2	3	4	5	0	1	2	3
Kachel 2:	INT_MAX	INT_MAX	0	0	1	2	3	4	0	1	2	3	4	0
Kachel 3:	INT_MAX	INT_MAX	INT_MAX	INT_MAX	INT_MAX	0	0	1	2	0	1	2	0	1

Adjazenztafel der 'Älter als'-Relation $\preccurlyeq$

- $\preccurlyeq$ kann als Adjazenzmatrix $\mathbf{Adj}_{\preccurlyeq}$ dargestellt werden
- $j \preccurlyeq i \equiv \mathbf{Adj}_{\preccurlyeq}[i,j] = 1$, ansonsten gilt $\mathbf{Adj}_{\preccurlyeq}[i,j] = 0$
- Offensichtlich muss $\preccurlyeq$ reflexiv sein: Für die Diagonale $d_0 \cdots d_{n-1} = 1$
- Außerdem ist $\preccurlyeq$ antisymmetrisch: $\mathbf{Adj}_{\preccurlyeq}[i,j] = 0 \Leftrightarrow \mathbf{Adj}_{\preccurlyeq}[j,i] = 1$
- Statt $\forall j \in \{k \in K | j > k\}$ wird für ein festes k hier $\forall j > k$ geschrieben

Adjazenztafel der 'Älter als'-Relation (Beispiel)



$_i{}^j$	0	1	2	3	4	5	6	7
0	d_0	0	0	0	0	0	0	0
1		d_1	0	0	0	0	1	0
2			d_2	0	1	1	1	0
3				d_3	1	1	1	0
4					d_4	0	1	0
5						d_5	1	0
6							d_6	0
7								d_7

LRU-Algorithmus mit Adjazenzmatrix

- Wurde unmittelbar vor dem t-ten Zeitschritt die Kachel $k \in K$ aufgerufen, so muss k unter $\preccurlyeq_t$ als $\preccurlyeq$ bzw. $\prec$ die folgenden Eigenschaften erfüllen:
  - $\forall j > k.\ \mathrm{Adj}_{\preccurlyeq}[k, j] = 0$ bzw. $\forall j > k.\ k \prec j$: Zeile auf Null setzen
  - $\forall i < k.\ \mathrm{Adj}_{\preccurlyeq}[i, k] = 1$ bzw. $\forall i < k.\ k \preccurlyeq i$: Spalte auf Eins setzen
- Der Älteste Eintrag zum Zeitpunkt t wird $x \coloneqq \max_{\preccurlyeq_t} K$ genannt:
  - $\forall i < x.\ \mathrm{Adj}_{\preccurlyeq}[i, x] = 0$ bzw. $\forall i < x.\ i \prec x$: Spalte ist Null
  - $\forall j > x.\ \mathrm{Adj}_{\preccurlyeq}[x, j] = 1$ bzw. $\forall j > x.\ j \preccurlyeq x$: Zeile ist Eins

  } $\forall j \in K.\ j \preccurlyeq x$

Berechnung des ältesten Elements (Aufgabe)

i \ j	0	1	2	3	4	5	6	7
0	d_0	1	1	1	0	1	0	1
1		d_1	0	1	0	0	0	0
2			d_2	1	0	0	0	1
3				d_3	0	0	0	0
4					d_4	1	0	1
5						d_5	0	1
6							d_6	1
7								d_7

Berechnung des ältesten Elements (Lösung)

i \ j	0	1	2	3	4	5	6	7
0	d_0	1	1	1	0	1	0	1
1		d_1	0	1	0	0	0	0
2			d_2	1	0	0	0	1
3				d_3	0	0	0	0
4					d_4	1	0	1
5						d_5	0	1
6							$\boldsymbol{d_6}$	1
7								d_7

Was ist hier das älteste Element?

i \ j	0	1	2	3	4	5	6	7
0	d_0	1	1	1	1	1	1	1
1		d_1	1	1	0	0	1	0
2			d_2	0	0	0	1	0
3				d_3	0	0	1	0
4					d_4	1	1	1
5						d_5	1	1
6							d_6	0
7								d_7

Was ist hier das älteste Element? (Lösung)

i \ j	0	1	2	3	4	5	6	7
0	$\boldsymbol{d_0}$	1	1	1	1	1	1	1
1		d_1	1	1	0	0	1	0
2			d_2	0	0	0	1	0
3				d_3	0	0	1	0
4					d_4	1	1	1
5						d_5	1	1
6							d_6	0
7								d_7

Beispiele zu Kachelaufrufen

i \ j	0	1	2	3	4	5	6	7
0	d_0	0	0	0	0	0	1	0
1		d_1	0	0	0	0	1	0
2			d_2	0	1	1	1	0
3				d_3	1	1	1	0
4					d_4	0	1	0
5						d_5	1	0
6							d_6	0
7								d_7

Beispiele zu Kachelaufrufen

i \ j	0	1	2	3	4	5	6	7
0	d_0	0	0	0	0	0	1	0
1		d_1	0	0	0	0	1	0
2			d_2	0	1	1	1	0
3				d_3	1	1	1	0
4					d_4	0	1	0
5						d_5	1	0
6							d_6	0
7								d_7

$k = 3$

i \ j	0	1	2	3	4	5	6	7
0	d_0	0	0	1	0	0	1	0
1		d_1	0	1	0	0	1	0
2			d_2	1	1	1	1	0
3				$\boldsymbol{d_3}$	0	0	0	0
4					d_4	0	1	0
5						d_5	1	0
6							d_6	0
7								d_7

Beispiele zu Kachelaufrufen

i \ j	0	1	2	3	4	5	6	7
0	d_0	0	0	0	0	0	1	0
1		d_1	0	0	0	0	1	0
2			d_2	0	1	1	1	0
3				d_3	1	1	1	0
4					d_4	0	1	0
5						d_5	1	0
6							d_6	0
7								d_7

$k = 3$

i \ j	0	1	2	3	4	5	6	7
0	d_0	0	0	1	0	0	1	0
1		d_1	0	1	0	0	1	0
2			d_2	1	1	1	1	0
3				$\boldsymbol{d_3}$	0	0	0	0
4					d_4	0	1	0
5						d_5	1	0
6							d_6	0
7								d_7

$k = 4$

i \ j	0	1	2	3	4	5	6	7
0	d_0	0	0	1	1	0	1	0
1		d_1	0	1	1	0	1	0
2			d_2	1	1	1	1	0
3				d_3	1	0	0	0
4					$\boldsymbol{d_4}$	0	0	0
5						d_5	1	0
6							d_6	0
7								d_7