Mathematical Model for the FIFO System

This document presents a mathematical model of a system where a traffic generator writes data into a FIFO and a consumer reads from it at different rates. MathJax is used to render the equations.

System Description

We consider a system with the following characteristics:

1. **Input (Producer):** A traffic generator produces data at a constant throughput `TP_in` (kB/μs) for a duration `T_dl` (μs). Over this duration, a total of `D` kB of data is inserted into the FIFO, where:

   $$D = T_{Pin} \cdot T_{dl}.$$

   The cumulative data inserted into the FIFO at time $t \geq 0$ is:

   $$I(t) = \begin{cases} T_{Pin} \cdot t & 0 \leq t \leq T_{dl}, \\ D & t > T_{dl}. \end{cases}$$

2. **Output (Consumer):** The consumer reads data from the FIFO at a constant rate T_{Pout1} until time t_0, and then switches to a higher rate T_{Pout2} for $t \geq t_0$. Since $T_{Pin} > T_{Pout1}$ and $T_{Pin} < T_{Pout2}$, initially the FIFO accumulates data because input exceeds output, and after time t_0, the FIFO empties because output exceeds input. The cumulative data read from the FIFO at time t is:

   $$O(t) = \begin{cases} T_{Pout1} \cdot t & 0 \leq t < t_0, \\ T_{Pout1} \cdot t_0 + T_{Pout2}(t - t_0) & t \geq t_0. \end{cases}$$

FIFO Occupancy

The FIFO occupancy $Q(t)$ at time t is given by the difference between the cumulative input and output:

$$Q(t) = I(t) - O(t).$$

We analyze the occupancy over time, depending on whether t_0 is less than or greater than T_{dl}.

Case A: $t_0 \leq T_{dl}$

For $0 \leq t < t_0$:

$$Q(t) = T_{Pin}t - T_{Pout1}t = t(T_{Pin} - T_{Pout1}).$$

Since $T_{Pin} > T_{Pout1}$, $Q(t)$ increases linearly until t_0. At $t = t_0$:

$$Q(t_0) = t_0(T_{Pin} - T_{Pout1}).$$

After t_0, the output rate changes to T_{Pout2}. For $t_0 \leq t \leq T_{dl}$:

$$Q(t) = T_{Pin}t - [T_{Pout1}t_0 + T_{Pout2}(t - t_0)].$$

Rearranging:

$$Q(t) = (T_{Pin} - T_{Pout2})t + (T_{Pout2} - T_{Pout1})t_0.$$

Since $T_{Pin} < T_{Pout2}$, the term $(T_{Pin} - T_{Pout2})$ is negative, hence $Q(t)$ decreases after t_0. Thus, the maximum occupancy in this case occurs at $t = t_0$:

$$\max Q = Q(t_0) = t_0(T_{Pin} - T_{Pout1}).$$

Case B: $t_0 > T_{dl}$

For $0 \leq t \leq T_{dl}$:

$$Q(t) = t(T_{Pin} - T_{Pout1}),$$

which increases linearly until $t = T_{dl}$. At $t = T_{dl}$:

$$Q(T_{dl}) = T_{dl}(T_{Pin} - T_{Pout1}).$$

Since no more data arrives after T_{dl}, and $t_0 > T_{dl}$, the FIFO starts to empty with output rate T_{Pout1} until t_0. During $T_{dl} < t < t_0$:

$$Q(t) = D - T_{Pout1}t,$$

which is decreasing. Therefore, the maximum occupancy in this scenario is at $t = T_{dl}$. Recall $D = T_{Pin}T_{dl}$:

$$Q(T_{dl}) = T_{dl}(T_{Pin} - T_{Pout1}) = D\left(1 - \frac{T_{Pout1}}{T_{Pin}}\right).$$

Optimal FIFO Size

The optimal FIFO size must be at least the maximum occupancy to prevent overflow. Thus, we have: - If $t_0 \leq T_{dl}$:

$$\text{FIFO}_{\max} = t_0(T_{Pin} - T_{Pout1}).$$

- If $t_0 > T_{dl}$:

$$\text{FIFO}_{\max} = D\left(1 - \frac{T_{Pout1}}{T_{Pin}}\right).$$

Summary

The optimal FIFO size depends on whether the output rate increases before or after the end of the input data duration:

- For `t₀ ≤ Tdl`:

$$\text{FIFO}_{\max} = t_0(T_{Pin} - T_{Pout1}).$$

- For `t₀ > Tdl`:

$$\text{FIFO}_{\max} = D\left(1 - \frac{T_{Pout1}}{T_{Pin}}\right).$$

This result shows how the maximum FIFO size scales with the delay before output rate changes, the total data volume, and the input/output rates.

Visualization

You can use the following Python code (with Plotly in dark mode) to visualize the occupancy and verify these results:

```
import numpy as np
import plotly.graph_objs as go
from plotly.offline import plot

# Parameters (example values)
TPin = 2.0      # kB/µs
TPout1 = 1.0    # kB/µs
TPout2 = 3.0    # kB/µs
D = 1000.0      # kB total data
Tdl = D / TPin   # µs
t0 = 300.0       # µs

t_max = max(t0, Tdl) + 200
t = np.linspace(0, t_max, 1000)
Q = np.zeros_like(t)

for i, time in enumerate(t):
    # Inflow
    if time <= Tdl:
        I = TPin * time
    else:
        I = D

    # Outflow
    if time < t0:
        O = TPout1 * time
    else:
        O = TPout1 * t0 + TPout2 * (time - t0)

    Q[i] = I - O

if t0 <= Tdl:
    fifo_max = t0 * (TPin - TPout1)
```

```
else:
    fifo_max = D * (1 - TPout1/TPin)

layout = go.Layout(
    template="plotly_dark",
    title="FIFO Occupancy Over Time",
    xaxis=dict(title="Time (μs)"),
    yaxis=dict(title="FIFO Occupancy (kB)"),
)

trace = go.Scatter(x=t, y=Q, mode='lines', name='Q(t)')
max_line = go.Scatter(
    x=[0, t_max],
    y=[fifo_max, fifo_max],
    mode='lines',
    line=dict(dash='dot', color='red'),
    name='Max FIFO'
)

fig = go.Figure(data=[trace, max_line], layout=layout)
fig.show()
```

This visualization confirms the derived formula for the maximum FIFO size under the given conditions.