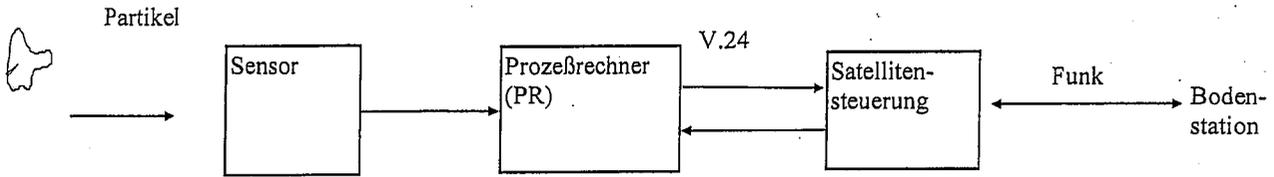


2. Aufgabe: Satellitenexperiment - Staubzähler - (25 P.)

Im Weltraum sollen permanent Staubpartikel mittels eines gegebenen Sensors in einem Satelliten gezählt werden. Der in dieser Aufgabe betrachtete **Prozeßrechner (PR)** des Staubzähl-Experiments im **Satelliten** hat die Aufgabe, die durch den Sensor gewonnenen **Meßdaten zu erfassen, sie auszuwerten** und alle **Ergebnisse auf Anfrage an die Bodenstation auf der Erde zu schicken**. Zu Veranschaulichung dient das folgende Systemschaubild:



Die **Bodenstation** ist der Master der **den PR über entsprechende Ein-Byte-Kommandos fernsteuert**. Diese Kommandos werden im Satelliten zunächst durch die Satellitensteuerung entgegengenommen und mittels serieller Verbindung (V.24) an den PR weitergereicht. Umgekehrt sendet der PR eine entsprechende Antwort an die Satellitensteuerung, die diese dann an die Bodenstation funkt. Diese serielle Verbindung wird mit der Geschwindigkeit von 1000 Bytes/s betrieben. Der eingesetzte PR ist mit Realzeitbetriebssystem OS9000 mit Prioritätsscheduling ausgerüstet. Das Senden und der Empfang eines Bytes kann gleichzeitig geschehen. Das Abholen des empfangenen Bytes und das Bereitstellen des zu sendenden Bytes kostet je 1ms Verarbeitungszeit. Der Empfang eines Bytes wird durch das Setzen des Events **BYTE_IN =1** signalisiert, und nach Abholen des Bytes automatisch rückgesetzt. Es darf das nächste Byte zum Senden übergeben werden, wenn das vorherige gesendete Byte vollständig übertragen wurde; was durch das Bit **BYTE_OUT=1** angezeigt wird. Es wird zunächst vorgeschlagen die auszuführenden Aufgaben auf die folgenden Tasks zu verteilen:

- Task A:** Bei bestehendem Funkkontakt der Satellitensteuerung mit der Erde wird diese Task A alle 10 Sekunden von Task C aktiviert ansonsten ruht sie: Die Task A ist **Sende- und Empfangstask in Einem**. Sie wartet auf ein Kommando, interpretiert es und führt es aus. Nach Erhalt und Interpretation des Kommandos schickt sie sofort eine Quittungsnachricht (Länge 1 Byte). Bei Kommando **MESSW** folgen dann noch alle vorliegenden Meßdaten (zwischen 1 - 100 Meßwerte je 16 Byte). Bei Kommando **RESET** muß sich der Prozeßrechner **sofort** (innerhalb von 20 ms) zurücksetzen. Die Interpretation eines beliebigen Kommandos Task A dauert maximal 1 ms. Nach 6 Sekunden beendet sich die Task A in jedem Fall.
- Task B:** Der intelligente Meßwertaufnehmer (Sensor) liefert alle 5 Minuten einen Satz von Meßwerten. Diese Meßwerte werden zunächst von Task B entgegengenommen und ausgewertet. Es entstehen so alle fünf Minuten zwischen 1 Meßwert (16 Byte) und 100 Meßwerten (1600 Byte). Die Ergebnisse dieser Auswertung werden dann in einem reservierten Speicherbereich (**shared-memory**) abgelegt. Von dort werden sie von Task A abgeholt. Die Task B benötigt in allen Fällen eine maximale Rechenzeit von 1 Sekunde und darf jederzeit unterbrochen werden.
- Task C:** Die Task C dient als Timertask und muß außerdem zyklisch wiederkehrend System-Selbsttestaufgaben durchführen. Task C wird bei Bedarf jede ms geweckt und ist aktiv. Task C erhöht alle 100ms zunächst einen Zeitzähler; dazu benötigt sie 1ms Rechenzeit. Nach Ablauf jeder Sekunde muß Task C innerhalb 100ms zusätzlich Diagnoseaufgaben: Test auf Funkkontakt zur Erde, System-Selbsttest und Watchdogreset durchführen; dazu benötigt sie 20ms Rechenzeit. Bei Funkkontakt zur Erde startet sie alle 10 Sekunden die Task A; dafür benötigt sie 10ms Rechenzeit.

Aufgabe 2.1: Wie groß ist jeweils die Auslastung je verursacht durch Task A, B und C (Rechenweg muß ersichtlich sein!!) (5 P.)

$$A = \frac{2[ms] + 2[ms] + 1600 \cdot 2[ms]}{10000[ms]} \quad B = \frac{1[sec]}{5 \cdot 60[sec]} \quad C = \frac{10[ms] + 100 \cdot 1[ms] + 10 \cdot 20[ms]}{10000[ms]}$$

Für Aufgabe 2.2: Es sei Funkkontakt und es fällt zunächst alle 5 Minuten nur 1 Meßwert an.

Aufgabe 2.2: Teilen Sie für diesen Fall den drei Tasks A, B und C Prioritäten (0=hohe, 1=mittlere, 2=niedrige Priorität) zu und begründen Sie kurz Ihre Entscheidung (2 P.)

	Priorität	Begründung
TASK A	hoch	Reset Zeit 20ms
TASK B	niedrig	Unterschied alle 5 min
TASK C	mittel	TIMER, aber braucht für Selbsttest 20ms / Reset