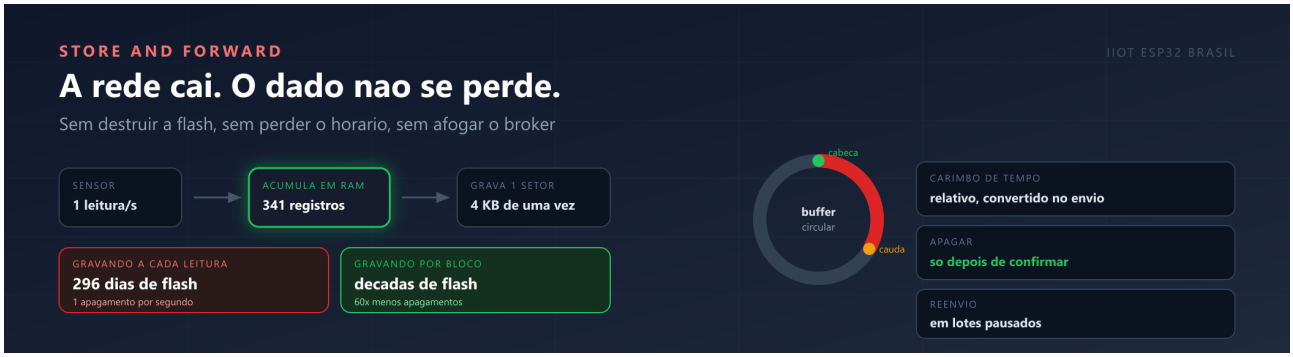
Buffer local store-and-forward

A rede cai, o dado não se perde - manual prático



Placa	ESP32 ou ESP32-S3
Framework	Arduino Core, via PlatformIO
Sistema de arquivos	LittleFS, em partição própria
Registro	12 bytes de tamanho fixo; 341 por setor de 4 KB
Nível	Avançado
Tutorial online	iiot.esp32.tec.br/tutoriais/buffer-local-quando-a-rede-cair

As contas de desgaste deste manual são **estimativas de ordem de grandeza** a partir do número usual de cem mil ciclos por setor, e não medições em bancada. O firmware do pacote também não foi validado em hardware: teste antes de usar em instalação que importe.

1. O dado existe, e se perde

Um gateway que lê um medidor a cada dez segundos e publica na nuvem funciona bem - até a internet cair. Aí, durante quarenta minutos, ele continua lendo e continua jogando fora. Quando a rede volta, o histórico daquele período simplesmente não existe.

Para um painel de temperatura, é chato. Para medição de energia faturável, para rastreabilidade de processo ou para investigar por que uma máquina parou às três da manhã, é inaceitável.

Buffer em RAM não basta

A primeira tentação é guardar num vetor na memória: rápido, simples, e não desgasta nada. É se perde no reinício.

Isso seria tolerável se reinício fosse raro, mas a queda de rede e o reinício andam juntos com uma frequência incomoda: falta de energia derruba os dois, o watchdog reinicia quando algo trava, e uma atualização OTA reinicia por definição. O buffer em RAM tende a evaporar exatamente na hora em que importava.

2. Desgaste da flash

Esta é a restrição que separa um projeto que dura de um que morre em meses.

A flash do ESP32 suporta na ordem de cem mil ciclos de apagamento por setor, e a granularidade é o setor inteiro de 4 KB - mesmo que você tenha mudado doze bytes.

Estrategia	Apagamentos	Vida útil estimada
1 registro/s, gravando a cada leitura, 1 setor	1 por segundo	cerca de 28 horas
idem, com desgaste distribuido em 1 MB	1 por segundo	cerca de 296 dias
acumulando em RAM e gravando 1x por minuto	1 por minuto	decadas

A solução não é gravar menos dados: é gravar **com menos frequência**. Com registros de 12 bytes, 341 cabem num setor de 4 KB. Gravando o bloco de uma vez, o desgaste cai por um fator de sessenta.

> O preço é perder até um minuto de dados num reinício abrupto. Compare com destruir a flash em dez meses e a escolha fica fácil. Se o seu caso não tolera nem isso, o caminho é FRAM externa, que suporta ordens de grandeza mais ciclos.

Registro fixo, não linha de JSON

Linha de JSON por leitura é conveniente e é armadilha por três motivos: o arquivo cresce sem limite, porque não há como sobrescrever a linha mais antiga; não da para saltar direto ao registro N; e o JSON gasta de três a cinco vezes mais bytes, o que multiplica o desgaste.

```
struct __attribute__((packed)) Registro {
uint32_t millisLeitura;
uint16_t sessao;
float valor;
uint8_t origem;
uint8_t estado;
};
```

Doze bytes exatos. O packed importa: sem ele o compilador insere preenchimento e o tamanho deixa de ser o que você calculou.

3. O carimbo de tempo

Aqui está a falha que só aparece depois que tudo parece funcionar.

O dado guardado offline precisa carregar **o instante em que foi lido**, não o instante em que foi enviado. Se o servidor carimba na chegada, seis horas de buffer viram seis horas de leituras com o mesmo horário, e o histórico fica inútil.

Só que o ESP32 não tem relógio ao ligar: descobre a hora por NTP, que exige a rede - a mesma que caiu. O cenário completo é falta de energia, religa sem rede, começa a ler e a bufferizar sem saber que horas são.

A saída

```
guarde millis() no registro, junto de um contador de sessao
quando o NTP sincronizar, calcule o instante da sincronizacao
instante = referencia - (millisAgora - millisDoRegistro)
```

O contador de sessão evita misturar leituras de antes e depois de um reinício, cujos valores de millis não são comparáveis. Registros de sessão anterior são marcados como hora desconhecida, em vez de receberem um horário inventado.

> É trabalho a mais. Sem ele, o buffer entrega dados com horário errado - o que costuma ser pior que não entregar.

4. Descarte, confirmação e reenvio

Descartar o mais velho ou o mais novo?

Tipo de dado	O que descartar	Por que
Telemetria continua	o mais antigo	se ficou seis horas fora e o buffer guarda duas, você quer as duas últimas
Eventos e alarmes	nada; sinalize quando encher	um alarme não pode sumir porque veio telemetria depois dele

O certo são dois buffers separados. Misturar os dois no mesmo anel garante que, no pior momento, você perde justamente o registro que explicava o problema.

Só apague depois de confirmar

Reenviar e apagar em seguida parece natural, e perde dado: se a conexão cair entre o envio e a confirmação, o registro foi apagado e nunca chegou.

A ordem certa é enviar, esperar a confirmação do broker, e só então avançar a cauda. Com MQTT isso significa qualidade de serviço 1 no mínimo, porque o nível 0 não confirma nada.

> E significa aceitar que o mesmo registro pode chegar duas vezes. **É melhor entregar duas vezes do que zero vezes:** deduplicar no servidor é trivial, recuperar dado perdido é impossível.

Reenvio sem afogar a rede

Quando a rede volta com horas de buffer, despejar tudo de uma vez quebra três coisas: o broker aplica limite de taxa e derruba a conexão; o laço fica preso no reenvio e para de ler o sensor, criando um novo buraco; e o consumo de pico pode provocar brownout.

Envie no máximo N registros por volta do laço, intercalados com o trabalho normal. Vinte por volta, com o laço rodando dez vezes por segundo, escoam duzentos por segundo sem travar nada.

5. Dimensionamento

A autonomia do buffer é especificação, não detalhe: informe a quem opera.

Capacidade	Intervalo de leitura	Autonomia aproximada
20.000 registros	10 segundos	cerca de 55 horas
20.000 registros	60 segundos	cerca de 13 dias
43.000 registros (512 KB)	10 segundos	cerca de 5 dias
43.000 registros (512 KB)	60 segundos	cerca de 30 dias

Para calcular o seu: divida o tamanho da partição pelo tamanho do registro para saber quantas leituras cabem, e multiplique pelo intervalo de leitura.

6. Roteiro de teste

Teste	Como fazer	Resultado esperado
Persistência	gerar registros, aguardar gravação e reiniciar	ao voltar, encontra o índice e continua de onde parou
Giro do anel	reduzir a capacidade, encher e continuar gravando	sobrescreve os mais antigos sem crescer o arquivo
Reenvio	acumular com o broker parado e depois subir	escoa em lotes, intercalado com o trabalho normal
Queda no meio	derrubar o broker durante o escoamento	para sem perder o não confirmado, e retoma depois
Contagem final	comparar entregues com gerados	os números batem; é o teste que prova a implementação

7. Solução de problemas e checklist

Sintoma	Causa provável	O que fazer
A flash morreu em poucos meses	gravação por leitura	acumule em RAM e grave um setor de cada vez
O arquivo cresce até encher	anexando em vez de anel	registro fixo e sobrescrita do mais antigo
Buffer volta vazio após reinício	índice não persistido	grave o índice junto do bloco, com sincronização
Todos os dados com o mesmo horário	carimbo na chegada	grave tempo relativo e converta no envio
Faltam registros após queda no reenvio	apaga antes de confirmar	avance a cauda só depois do reconhecimento
Broker derruba quando a rede volta	reenvio sem limite	envie em lotes pausados
Sensor para de ser lido no reenvio	laço preso no escoamento	limite quantos registros saem por volta
Perdi um alarme e sobrou telemetria	buffers misturados	separe telemetria de eventos

Checklist final

	Item
[ ]	Partição de sistema de arquivos declarada no particionamento
[ ]	Registro de tamanho fixo com atributo packed
[ ]	Acumulo em RAM até encher um setor antes de gravar
[ ]	Gravação também por tempo máximo, para sensor lento
[ ]	Índice persistido junto do bloco, com marca de validade
[ ]	Buffer circular sobrescrevendo o mais antigo
[ ]	Telemetria e alarmes em buffers separados
[ ]	Tempo relativo no registro, com contador de sessão
[ ]	Conversão para horário real no momento do envio
[ ]	Cauda avança só após confirmação do broker
[ ]	Campo de sequência para deduplicação do outro lado
[ ]	Reenvio limitado por lote a cada volta do laço
[ ]	Autonomia calculada e informada a quem opera
[ ]	Teste de queda no meio do reenvio: contagem final bateu